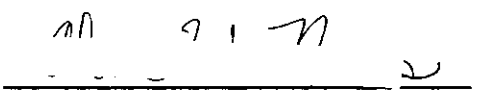
In presenting the dissertation as a partial fulfillment of the requirements for an advanced degree from the Georgia Institute of Technology, I agree that the Library of the Institution shall make it available for inspection and circulation in accordance with its regulations governing materials of this type. I agree that permission to copy from, or to publish from, this dissertation may be granted by the professor under whose direction it was written, or, in his absence, by the Dean of the Graduate Division when such copying or publication is solely for scholarly purposes and does not involve potential financial gain. It is understood that any copying from, or publication of, this dissertation which involves potential financial gain will not be allowed without written permission.



AN INVESTIGATION OF THE EFFECTS OF NON-PREEMPTIVE PRIORITY AND OPERATOR INTERFERENCE IN A TEXTILE WEAVING PROCESS

A THESIS

Presented to

The Faculty of the Graduate Division

by

Alan Wayne Nass

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Industrial Engineering

Georgia Institute of Technology

February, 1965

AN INVESTIGATION OF THE EFFECTS OF

NON-PREEMPTIVE PRIORITY AND OPERATOR

INTERFERENCE IN A TEXTILE WEAVING PROCESS

Approved:

Chairman

Date approved by Chairman: March 9, 1965

ACKNOWLEDGMENTS

Simulation of queueing problems is often discussed in the literature, but very little has been published regarding how this simulation should be done. Dr. Joseph J. Moder encouraged the author to examine this problem after the study was suggested to the author by the people at Callaway Mills.

Dr. William W. Hines, the thesis advisor, deserves special credit for his help in further developing the study. His assistance is greatly appreciated.

Special credit is also given to Dr. D. E. Fyffe and Professor R. K. Flege for their suggestions on the manuscript and to Mr. Walter Fleming of the Rich Electronic Computer Center for his aid in the computer programming.

Personal appreciation is given to my wife, Connie, for her patience and understanding during my graduate work.

TABLE OF CONTENTS

Page

ACKNOWLEDGMENTS ii

LIST OF TABLES. iv

LIST OF ILLUSTRATIONS v

SUMMARY 1

Chapter

I. INTRODUCTION 3

Objective
Purpose
Definition of the Problem
Scope and Limitations
Assumptions

II. LITERATURE SURVEY. 13

Important Theorems
Priority

III. EXPERIMENTAL PROCEDURE 22

Monte Carlo Simulation
Computer Program
Comments

IV. RESULTS. 30

Definition of Terms
Discussion of Results

V. CONCLUSIONS AND RECOMMENDATIONS. 43

Conclusions
Recommendations

APPENDIX. 46

BIBLIOGRAPHY. 65

LIST OF TABLES

Table		Page
1.	Comparison of Queue Disciplines for Phase I.	33
2.	Comparison of Queue Disciplines for Phase II	34
3.	Comparison of Queue Disciplines for Phase III.	35
4.	Cost Comparisons of Queue Disciplines for Phase I.	36
5.	Cost Comparisons of Queue Disciplines for Phase II	37
6.	Cost Comparisons of Queue Disciplines for Phase III.	38
7.	Measures of Effectiveness for Phase I.	39
8.	Measures of Effectiveness for Phase II	40
9.	Measures of Effectiveness for Phase III.	41
10.	Summary of Optimum Cost Data for the Entire System	42

LIST OF ILLUSTRATIONS

Figure — Page

1. Flow Diagram of Queueing Processes. 9
2. Flow Diagram for Warp Change 1. 10
3. Flow Diagram for Warp Change 2. 11
4. Flow Diagram for Warp Change 3. 12

SUMMARY

The maintenance of looms in the textile weaving process is similar to certain queueing models for machine maintenance. A loom has to be set up after a warp-out, and this operation requires a sequence of repairs by different servicemen before the loom can begin production again. A loom also requires different types of service after it is running but these repairs are handled by only one type of serviceman.

For the purposes of this study, the two classifications of repairs are considered as two separate queueing problems. This study is concerned with the first type of queueing process and examines the effects of two different queue disciplines. The first discipline is one of first-come-first-served. The second queue discipline is one of priority. The priority of an arrival is based upon the arrival's mean service time divided by its cost of queueing for a unit of time. The arrival with the minimum priority value is serviced first.

Data were gathered from an actual textile weaving process which wove essentially the same type of fabric and used in this study. Because the data did not follow the usual assumptions for queues in series, a simulation approach to the problem was employed. Analytical procedures require random arrivals and exponential service-time distributions. This queueing process violated the assumptions used in the existing literature on queues in series by having normal service-time distributions and multiple service rates at each repair station. Simulation was also needed because of the priority queue discipline.

The simulation was done on a Burroughs B 5000 computer. One thousand arrivals were considered, and the number of repairmen at each service phase was varied. Seven different values of idle-time cost were used to compare the economical effects of the two different queue disciplines. The simulation program also prints out data to determine the maximum queueing time of the arrivals, the average queueing time for those arrivals who have to wait, the number of arrivals that did not wait for service, the mean percentage of time an operator is working, and the mean machine idle-time. The simulation program is found in Appendix A.

The results of the study showed that the priority queue discipline is slightly better than the queue discipline of first-come-first-served. However, the actual implementation of a priority queue discipline for the textile process under study would not be particularly beneficial because the cost of training the personnel and the book work required for keeping account of an arrival's priority would probably offset the savings gained by using the priority queue discipline.

CHAPTER I

INTRODUCTION

Objective

The objective of this study is to compare two different types of queue disciplines that may be used for loom maintenance in the textile weaving process. The first queue discipline for determining which arrival to service is commonly known as first-come-first-served. The second type involves the principle of priority and is based upon the value given by an arrival's mean service time divided by its cost of queueing for a unit time. If more than one arrival is waiting for service, then the one with the minimum priority value is serviced first.

Purpose

The purpose of the study is to determine the economical effects of using each type of queue discipline for that part of loom maintenance which requires a loom to pass through a sequence of repairs because of a warp change*. The loom maintenance system is composed of two types of queueing processes. In the first process, a loom breaks down and requires a series of several types of service before it can begin production. The second queueing process is similar to the first, except that only one type of service is needed before the loom can go back into operation. The entire system corresponds to setting up looms for produc-

* A warp change is defined as the replenishing by the new warp of the old warp which was depleted by the weaving process.

tion and then maintaining them while they are in operation.

Both alternatives are simulated for the process of queues in series, and their results are compared. The queueing process of maintaining running looms is not simulated, but a method for simulating this process is indicated.

Definition of the Problem

During the weaving process, a loom may stop production because of a warp change or because of certain minor breakdowns. The possibility of a major mechanical breakdown of the loom will be excluded. When a warp change occurs a certain sequence of repair phases is necessary in order for the loom to begin production again. The sequence of repair phases is dependent upon the type of warp change that is required. The minor breakdowns have a short service time and may be of three types. The first type occurs when there is a filling* break. The second occurs if there is a warp** break. The third type of minor breakdown is one of a mechanical nature, where no major repair of the loom is necessary.

The weaving process can therefore be broken down into two classifications of maintenance. The first classification requires that the loom be set up to begin another weaving process. The second classification of service is necessary to keep the loom running once it has been set up.

The number of repairmen necessary to keep production at a desired level depends upon how often the above classifications of repair arise.

---

* The yarn which interlaces with the warp yarn to form a woven fabric.

** The yarns that run lengthwise in a woven fabric.

If too many repairmen are utilized, then there will be excessive personnel idle time. However, if not enough repairmen are used then extra loom downtime occurs.

There is also another important factor to be considered. Delivery dates on certain types of cloth must be met. If the delivery of cloth is late, then a penalty cost or a loss of goodwill takes effect.

The weaving process is therefore similar to a machine maintenance problem where machines must be set up and kept running.

Brief History of the Problem

The queueing system approach was suggested by the author while studying the problem of starting and maintaining running looms in a textile weaving room. Repairs were handled on a first-come-first-served basis with a rule-of-thumb method for determining the number of repairmen. Suggestions were made that the looms might be serviced in a more economical manner.

The problem was similar to some which have been solved by the application of certain analytical queueing procedures for machine maintenance. An effort was made to see if the problem could be solved analytically; however, certain assumptions of the model made it too difficult to solve in this manner, and a simulation approach was employed. Simulation has the advantage that one can change certain parameters and structure elements of the model and compare different alternatives. Cox and Smith (4) derived a priority queue discipline for a single-server system, and simulation offered a chance to compare the priority queue discipline that they derived against one of first-come-first-served.

Scope and Limitations

This study is somewhat limited in general applications because it is the simulation of a particular textile weaving process. For example, the arrival and service times that are incorporated into the model are valid only to the particular system studied.

The basic structure of the simulation program could be adapted to fit the needs of a system which is similar to the one studied. Different arrival-time and service-time distributions could be employed, and if the other assumptions were the same as those for the model in this study, then the simulation program would prove helpful. The logic of certain parts of the program could also be used in formulating many different types of queueing processes if none of the assumptions of the queueing process conflict with those of this model. It must be remembered that the results derived from this model are not necessarily true for other models formulated somewhat differently.

Assumptions

The queueing system under study has several assumptions, some of which conflict with those commonly found in the literature. These assumptions include random arrivals, truncated normal distributions for the service times, and different service rates for each operator. An analytical solution to the problem of queues in series is practically impossible to derive unless there is an assumption of exponential service-time distributions and only one service rate for the operators at a particular repair station. Therefore, simulation is necessary to compare the effects of a priority queue discipline. The second queueing process of maintaining

running looms is also complicated by the fact that an operator may have different service rates.

It has been assumed that workers can be trained to perform one of three types of services. The types of service are prepare, tie, and finish. At the present time there are more classifications of workers, but this study assumes that the advantages of training the workers to do similar types of repair would offset the cost of training and the increase in wage rates. The combining of similar jobs has been suggested by people in the textile field. For example, the warp hauler and blow-off man are combined to perform the task of preparing the warp for the tie machine tender. The tie machine tender does the function of tying the old warp to the new warp. The finish classification includes the tasks presently handled by the warp changer, smash hand, and the change hand.

For this study only three classifications of warp change on Draper Looms are examined. These are repeat warps with the same width or decrease in width, repeat warps with an increase in width, and a style change with a change of reeds*. It has been assumed that the first type of warp change occurs approximately 65 percent of the time, the second type change occurs 10 percent of the time, while the third type change arises the remaining 25 percent. Other types of warp changes are possible, but these three were selected as the most significant because they occur a large majority of the time.

A function common to the three types of warp change is that of tying the ends of the old warp beam to those of the new warp beam. There

---

*The series of parallel strips of wires which force the filling up to the web and separate the threads of the warp.

are three main ways of accomplishing the task of tying. The first method is to use a Zelweger-Uster tying machine. The second method is to use a Barber-Colman machine for tying the ends. And the third method is for the tie machine tender to hand tie the ends. Estimates indicate that the Zelweger-Uster tying machine is used approximately 50 percent of the time, while the Barber-Colman tying machine is used 25 percent of the time. Hand tying occupies the remaining 25 percent of the time. These percentages hold true regardless of the type of warp change that is performed.

The arrival rate of machines needing a warp change has been established from time studies to be approximately 11.6 per eight-hour shift. The mean service time and a minimum and maximum time for each type of service were found from time studies of the different repairmen. These mean service times and their respective ranges are indicated on Figure 1. The times were derived by combining the different times of the repairmen used in this study. Figure 2, Figure 3, and Figure 4 show the actual sequence of repairs for a warp change before certain types of repair were combined to form the cateogries of prepare, tie, and finish.

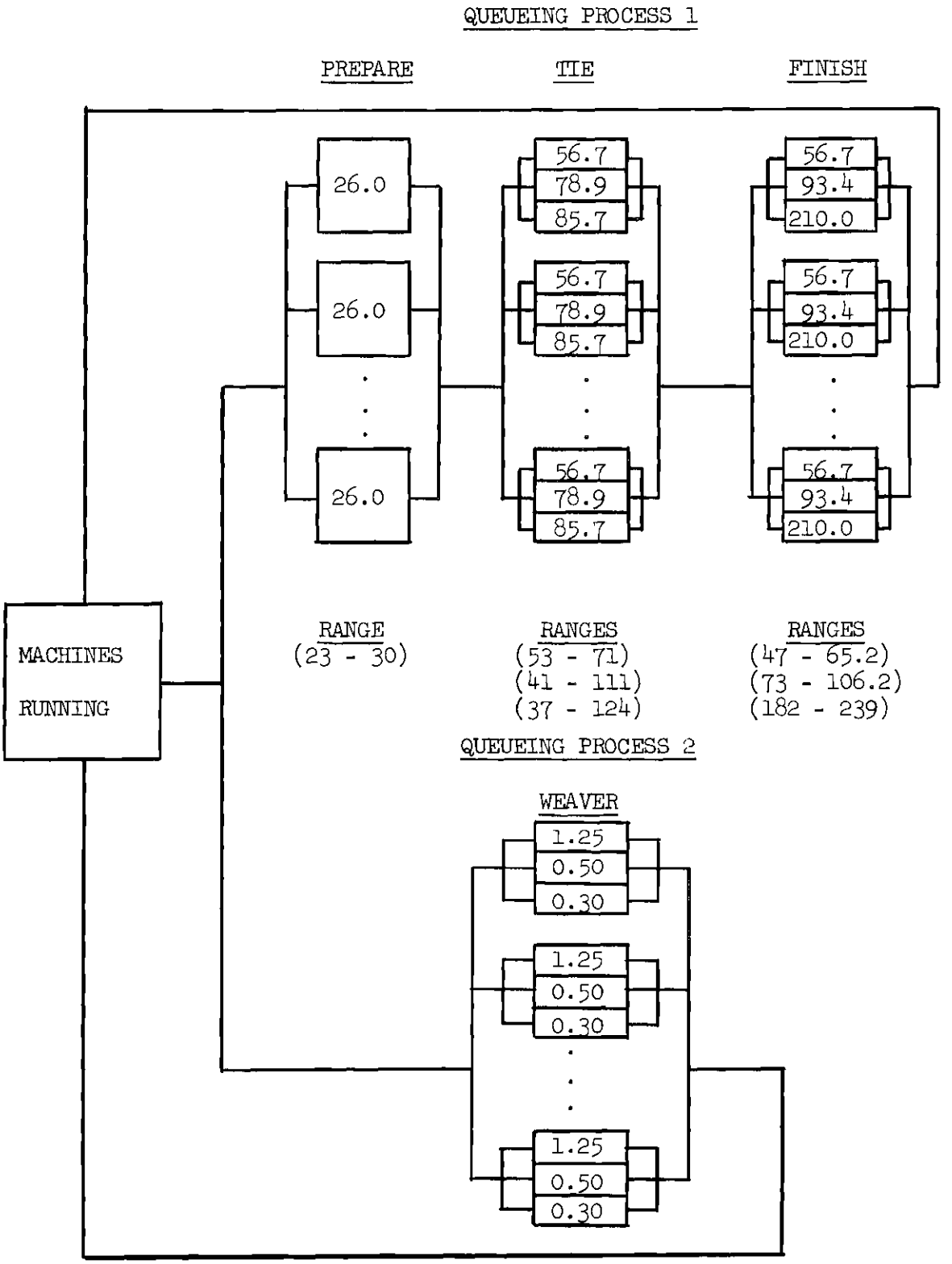


Figure 1. Flow Diagram of Queueing Processes

Draper Loom - Repeat Warps
Same Width or Decrease in Width

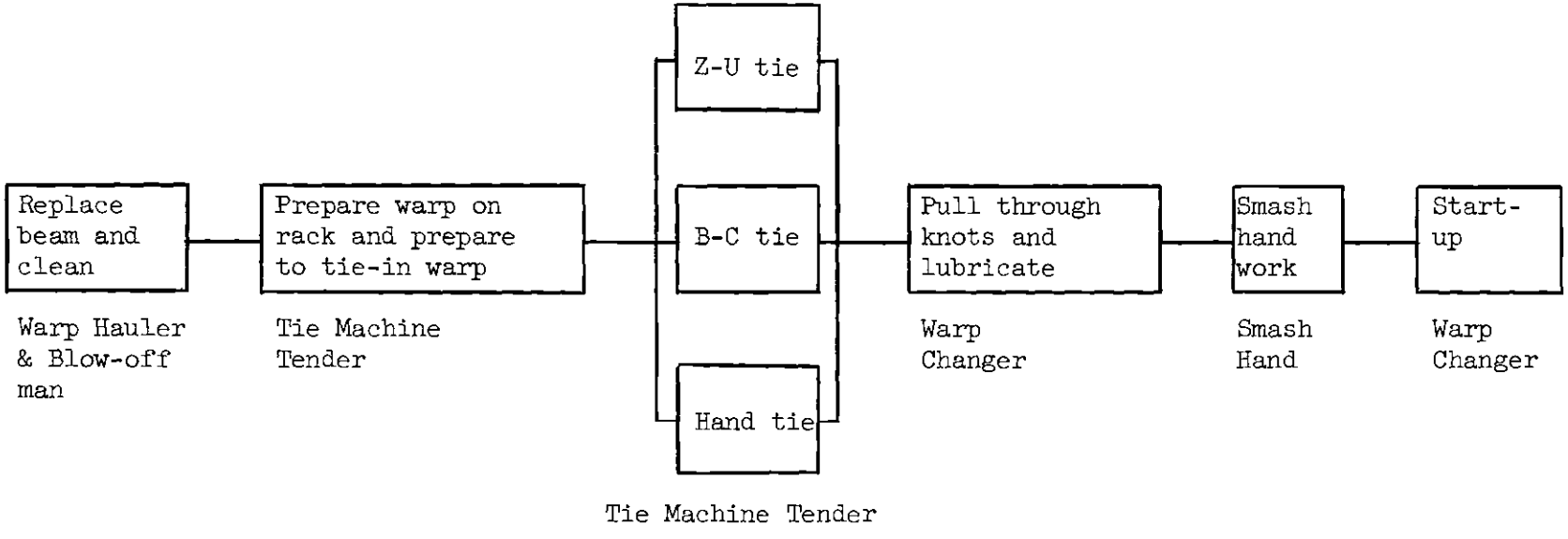


Figure 2. Flow Diagram for Warp Change 1

Draper Loom - Repeat Warps
Increase in Width

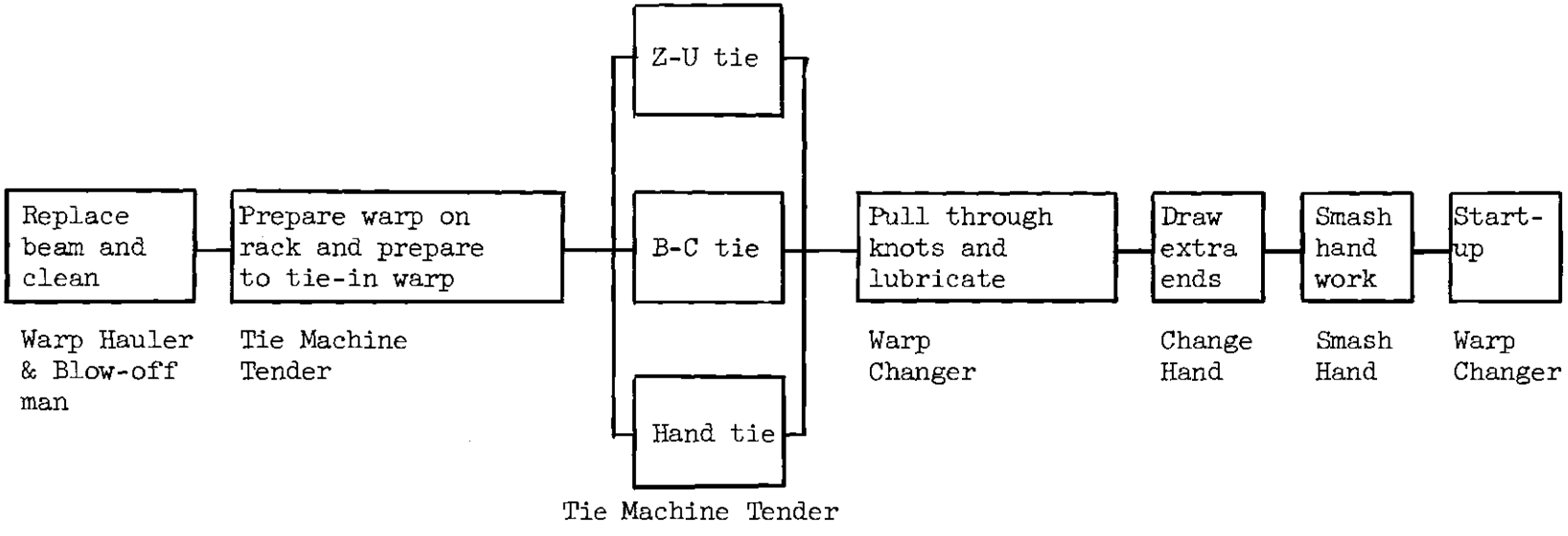


Figure 3. Flow Diagram for Warp Change 2

Draper Loom - Style Change
Change Reeds

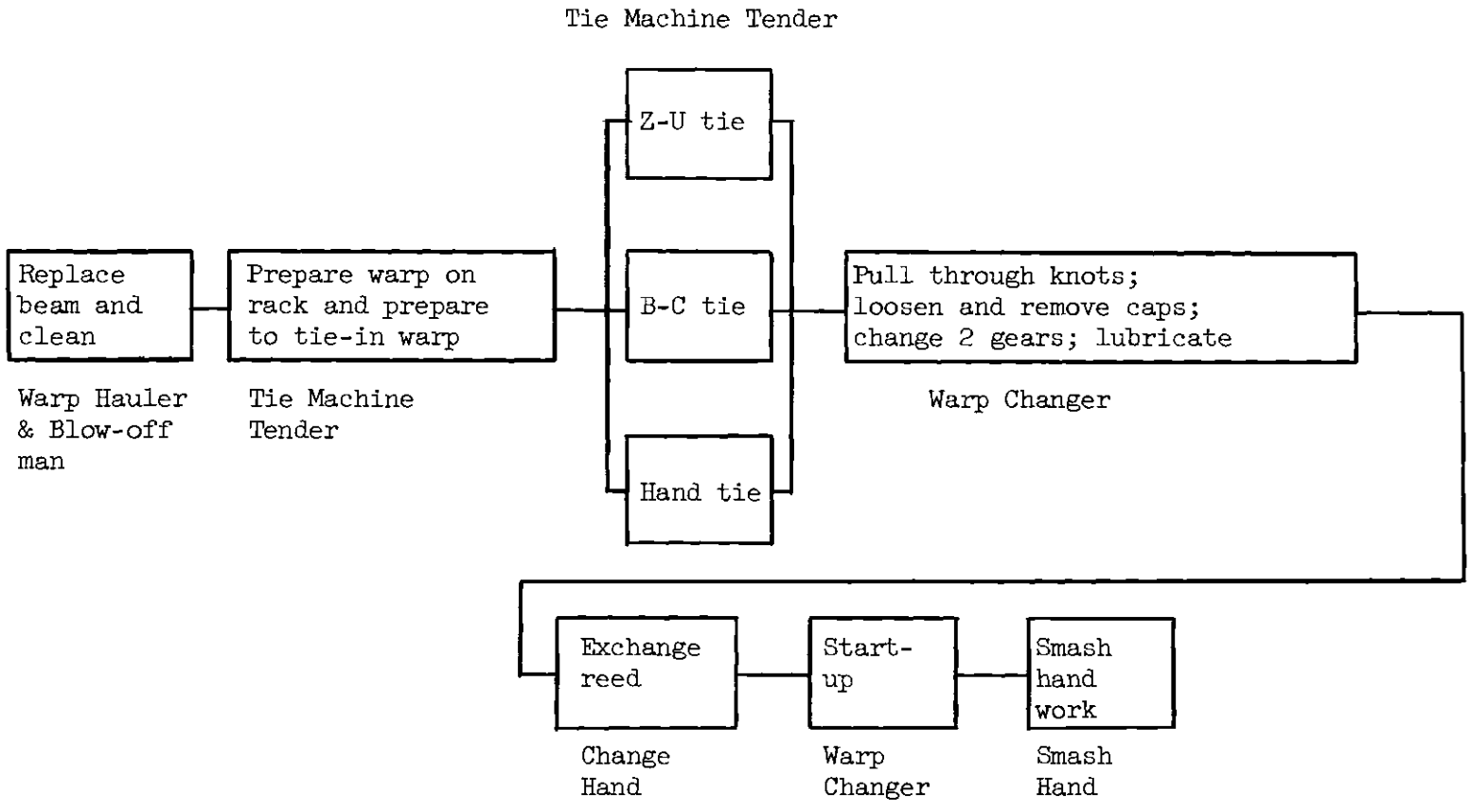


Figure 4. Flow Diagram for Warp Change 3

CHAPTER II

LITERATURE SURVEY

Important Theorems

The queueing system being studied is composed of two processes which have been discussed in the literature but with several different assumptions. The first queueing process corresponds to phase-type queueing and is dependent largely on several theorems discussed below.

The first theorem says that the steady-state output of a queue with c channels in parallel, with Poisson input and parameter λ and the same exponential-service-time distribution with parameter μ for each channel, is itself Poisson-distributed. Burke (1) has given a proof of this theorem. The second theorem states that if arrivals and departures of a single-channel queue are Poisson-distributed, the service-time distribution is exponential or a step function at zero. Reich (13) discusses the proof of this theorem. Finch (5) proved that an infinite calling population and exponential service are necessary and sufficient conditions for the independence of interdeparture intervals and the independence of the queue length left by a departing unit from the interval since the previous departure. Most of the work on queues in tandem or in series incorporates these theorems to arrive at certain results and measures of effectiveness.

For instance, R. R. P. Jackson (9) presented the queueing process where customers arrive at random and are served in order of arrival at each of a number of counters arranged in series. Each counter can have

many repairmen, and the service-time distribution at each counter is assumed to be negative exponential.

The steady state equations for the system are given by R. R. P. Jackson to be

$$(\lambda + \sum_{j=1}^{k} \delta[n_j] a[n_j] \mu_j)\, P\,(n_1, n_2, \cdots, n_k)$$

$$= \sum_{j=1}^{k} \delta[n_j + 1]\, a[n_j + 1]\, \mu_j\, P\,(n_1, n_2, \cdots, n_j + 1, n_{j+1} - 1, \cdots, n_k)$$

$$+ \lambda\, P(n_1 - 1, n_2, \cdots, n_k),$$

where all probabilities containing a negative value of n are automatically put equal to zero. The last term in the summation on the right-hand side involves a plus one term but no minus one term and

$$a[n_j] = n_j, \text{ for } \quad n_j < c_j$$

$$a[n_j] = c_j, \text{ for } \quad n_j \geq c_j$$

also,

$$\delta[n_j] = 1 \text{ , for } \quad n_j \neq 0$$

$$\delta[n_j] = 0 \text{ , for } \quad n_j = 0$$

$$\delta[n_{k+1}] = 1 \text{ .}$$

The terms are defined as follows:

c_j = number of repairmen at the j th phase

n_j = number of customers at the j th phase

μ_j = the service rate per repairman at the j th phase

λ = the arrival rate of the customers.

The steady-state probability of n_1 customers in the first phase of service, n_2 in the second phase, and continuing for n_k at the k^{th} phase is

$$P(n_1, n_2, \cdots, n_k) = P(0) \prod_{j=1}^{k} b[n_j],$$

$$b[n_j] = 1/n_j!(\lambda/\mu_j)^{n_j}, \text{ for } n_j < c_j$$

$$b[n_j] = 1/c_j!(\lambda/\mu_j c_j)^{n_j}(c_j)^{c_j}, \ n_j \geq c_j.$$

Since the sum over all n_j must equal unity, we have

$$\sum_{n_1=0}^{\infty} \cdots \sum_{n_k=0}^{\infty} \left[\prod_{j=1}^{k} b(n_j) \right] = \prod_{j=1}^{k} \sum_{j=0}^{\infty} b(n_j) \equiv \prod_{j=1}^{k} A_j, \ j = 1, \cdots, k$$

and

$$P(0) \equiv P(0, 0, \cdots, 0) = \prod_{j=1}^{k} A_j^{-1}$$

where

$$A_j = \sum_{n_j=0}^{\infty} b[n_j].$$

A proof of uniqueness of the above solution is also given by R. R. P. Jackson.

The marginal probability of the number of customers in the j^{th} phase is found by summing out over the number in all other phases.

That is

$$P_j(n) = \sum_{n_1=0}^{\infty} \sum_{n_2=0}^{\infty} \cdots \sum_{n_{j-1}=0}^{\infty} \sum_{n_{j+1}=0}^{\infty}$$

$$\cdots \sum_{n_k=0}^{\infty} P(n_1, n_2, \cdots, n_{j-1}, n, n_{j+1}, \cdots, n_k).$$

This result gives for the probability that there are n_j customers in the j^{th} phase of service

$$P_j(n) = \frac{b[n_j = n]}{A_j}$$

An important result is that the steady state probability distribution of customers in the j^{th} phase of service is dependent only upon the parameters λ, μ_j, and c_j. This is the same as the solution obtained for c channels in parallel with random arrivals, exponential service time, and a strict first-come-first-served queue discipline with an infinite queue size allowable. Therefore, measures of effectiveness for the system as a whole are given as the sum of the separate averages for the individual phases.

J. R. Jackson (8) considers a similar problem but allows Poisson arrivals from outside to enter different phases as well as those arrivals from the previous phase of service. He then gives an expression for the probability of a certain number of customers in each phase.

Hunt (7) used the same assumptions of Poisson input, exponential

service times, and sequential service at each repair phase beginning at the first phase, to examine the ratio of mean arrival rate to mean service rate for different sizes of queue lengths at the first phase and subsequent phases.

Priority

One can see that constant service times and a priority queue discipline are very difficult to study analytically and usually are solved only by simulation.

Cobham (3) was one of the first people to investigate the theory of priority. He derived a formula to express the mean waiting time for a customer of priority p, where there are r different types of priority. He studied the case where arrivals are random, and the repair facility has exponential service. These results were further expanded to include the case of c repair facilities with random arrivals and the same exponential service rate for each channel. Holley (6) later simplified the method of Cobham to arrive at the same results for the mean waiting time of a customer of priority p.

Phipps (11) published a paper following Cobham's article and treated the priorities as continuous. He considered only the case of one repair station with random arrivals from an infinite population of machines. The priority was given by the average time necessary to service a machine, the shorter the repair time the higher the priority. Phipps used continuous priorities so that he could better analyze Cobham's work and apply its results for machine maintenance.

Kesten and Runnenburg (10) have derived a formula for the Laplace

transforms of the waiting time distributions for the various priority classes. These transforms are used for the single channel case with random arrivals and independent, arbitrarily distributed service times.

In determining a priority system of queueing, several factors must be considered. If the downtime cost of a machine is particularly high, then it is reasonable to give this machine a higher priority than one with a lower cost of waiting. However, if the downtime cost of all machines is the same, then one would want to assign a higher priority to those machines with smaller service times in order to reduce the overall mean queueing time. Cox and Smith (4) have proposed a non-preemptive priority queue discipline which considers both of the above ideas.

One general-purpose service channel was studied where each machine was given a priority in a class from one through k. The number one denotes the highest priority and k denotes the lowest type of priority. Once a machine is being repaired, it remains there until it completes its service. Members of the same priority class are served on a first-come-first-served basis.

The machines of different classes are assumed to breakdown independently at rates $\lambda_1, \lambda_2, \cdots, \lambda_k$, and the unit time is chosen so that $\lambda_1 + \lambda_2 + \cdots + \lambda_k = 1$. Therefore, λ_j is the probability of an arrival of machine class j.

Let the service times of different machines be independently distributed with $B_j(x)$ as the distribution function for the machine of class j. The overall service-time distribution is given as

$$B(x) = \sum_{j=1}^{k} \lambda_j B_j(x).$$

Let b_j and c_j be the first and second moments, respectively, of $B_j(x)$. For example

$$b_j = \int_0^\infty x \, dB_j(x) \quad \text{and} \quad c_j = \int_0^\infty x^2 \, dB_j(x).$$

In the same manner b and c will be used for the moments of B(x) so that

$$b = \sum_{\text{all } j} \lambda_j b_j \quad \text{and} \quad c = \sum_{\text{all } j} \lambda_j c_j.$$

Then the traffic intensity ρ is equal to b since

$$\sum_{\text{all } j} \lambda_j = 1.$$

It is assumed that $\rho < 1$, and that the properties of the system have a stationary probability distribution.

Cox and Smith solved for certain important properties of this system and a method for assigning priorities. They found that Q_j, the mean queueing time of the j^{th} type machine, to be

$$Q_j = \frac{1/2 \, c}{(1 - \sum_{i=1}^{j-1} \lambda_i b_i)(1 - \sum_{i=1}^{j} \lambda_i b_i)}.$$

From Q_j, the mean queueing time, Q, of all machines is

$$Q = \sum_{\text{all } j} \lambda_j Q_j = \frac{c}{2} \sum_{j=1}^{k} \frac{\lambda_j}{(1 - \sum_{i=1}^{j-1} \lambda_i b_i)(1 - \sum_{i=1}^{k} \lambda_i b_i)}.$$

If there are k types of machines and the downtime cost for the j^{th} machine is constant and equal to w_j; then this implies that the downtime cost depends only upon the mean queueing time and that the mean cost is

$$C = \sum_{\text{all } j} \lambda_j w_j Q_j = \frac{c}{2} \sum_{j=1}^{k} \frac{\lambda_j w_j}{(1 - \sum_{i=1}^{j-1} \lambda_i b_i)(1 - \sum_{i=1}^{j} \lambda_i b_i)}.$$

To minimize the expression for C, we suppose that $k > 3$, and that in the expression for C we change all two's into three's and vice versa. A new cost C' is then found when the classes two and three have their priorities interchanged. The difference between the old and new cost is found to be

$$C - C' = \frac{c\Delta}{2} \left(\frac{w_2}{b_2} - \frac{w_3}{b_3}\right),$$

where

$$\Delta = (1 - \lambda_1 b_1 - \lambda_2 b_2)^{-1} + (1 - \lambda_1 b_1 - \lambda_3 b_3)^{-1}$$

$$- (1 - \lambda_1 b_1)^{-1} - (1 - \lambda_1 b_1 - \lambda_2 b_2 - \lambda_3 b_3)^{-1}.$$

It can be verified that $\Delta < 0$, so that $C < C'$ if and only if $b_2/w_2 < b_3/w_3$. Therefore, if $b_2/w_2 > b_3/w_3$ the mean cost can be reduced by changing the priority classifications of machine type two and machine type three.

In the general case, the same argument can be used to show that when

$$b_{j-1}/w_{j-1} > b_j/w_j,$$

the mean cost is reduced by changing the priority classification of the (j-1)-machine type and j-machine type. The result holds for cases of $j = 2, 3, \cdots, k$.

Therefore, this process can be used to reduce the mean cost by applying a priority system based on the quantity

$$\frac{\text{mean service time}}{\text{cost of queueing for unit time}},$$

where the lower the value, the higher the priority.

CHAPTER III

EXPERIMENTAL PROCEDURE

Monte Carlo Simulation

The Monte Carlo technique consists of examining a process by replacing the actual universe of items in the process by some probability distribution which is assumed to be theoretically equivalent to the actual universe of items. Then the theoretical population described by the chosen probability distribution is randomly sampled.

Monte Carlo methods are useful in examining queueing problems that are difficult to analyze mathematically. They also offer advantages over actual experimentation with the real system to determine what changes are produced by varying certain parameters of the system. The Monte Carlo methods can generate many weeks of data, and the parameters of the system can be manipulated to compare different alternatives.

a. Generation of Arrival Times: The first step in generating arrival-time intervals is to generate appropriately distributed random quantities. In order to produce observations independently distributed with distribution function F (X), the following procedure is utilized.

Let r_1, r_2, $\cdots$ be independently and uniformly distributed over the interval (0,1), and let X_i be the largest value of X for which $r_i \geq F(X_i)$, where i = 1, 2, $\cdots$. Then the X_i have the required distribution. When F (X) is a continuous strictly increasing function, the condition defining the X's may be replaced by the equation $r_i = F(X_i)$.

In this study we require the time between arrivals to be exponentially distributed with mean λ. Then $F(X) = 1 - e^{-\lambda X}$ for $X \geq 0$, and

$$X_i = - (1/\lambda) \ln (1 - r_i),$$

or equivalently,

$$X_i = - (1/\lambda) \ln (r_i),$$

since r_i and $1 - r_i$ have the same uniform distribution.

Arrival times are positioned on a real time scale where the time of the k^{th} arrival is equal to the sum of the preceding k arrival-time intervals that were generated.

b. Generation of Service Times: The generation of the service-time intervals is accomplished in a manner similar to the arrival times, but where the service-time intervals are normally distributed.

If N numbers, $r_1, r_2 \cdots, r_N$, are generated from a uniform distribution over the interval (0,1), then each r_j is uniformly distributed with mean 1/2 and variance 1/12.

Let $S = r_1 + r_2 + \cdots + r_N$. From the Central Limit Theorem we can say that S is approximately normally distributed with mean N/2 and variance N/12 if N is sufficiently large. Also by the Central Limit Theorem we know if

$$X = X_1' + X_2' + \cdots + X_N'$$

and the X_j' are independent and identically distributed with means $\tilde{\mu}$ and vari-

ances $\tilde{\sigma}^2$, then

$$\frac{X - N\tilde{\mu}}{\tilde{\sigma}\sqrt{N}}$$

is approximately normally distributed with a mean of zero and a variance of one if N is taken sufficiently large. In this particular study the value of N is 10. (12)

If we wish X to be normally distributed with mean μ and variance σ^2, the above results can be used to establish the relation,

$$X = \left[\frac{\sum_{i=1}^{N} r_i - (0.5)\,N}{\sqrt{N/12}}\right] \cdot \sigma + \mu.$$

The transformation changes S to a variable X which is normally distributed with a mean of μ and a variance of σ^2.

c. Generation of Random Numbers: Random numbers are generated by the congruential multiplicative method. This method works in the following manner. Let,

RI_o = an initial 10 digit random integer,

c = a constant 11 digit integer,

R_i = the i^{th} random number generated,

Modulus 10^M = an operation which preserves only the last M least significant digits of a product.

Then,

$$RI_1 = c \cdot RI_0 \text{ (Modulus } 10^M); \quad R_1 = 10^{-M} \cdot RI_1;$$

$$RI_2 = c \cdot RI_1 \text{ (Modulus } 10^M); \quad R_2 = 10^{-M} \cdot RI_2;$$

and

$$RI_k = c \cdot RI_{k-1} \text{ (Modulus } 10^M); \quad R_k = 10^{-M} \cdot RI_k.$$

Computer Program

A Burroughs B 5000 computer (2) was used for the simulation study. The B 5000 computer has almost unlimited capacity with approximately 32,000 cells of storage and is ideally suited to handle large simulation studies. The program was written in the Burroughs Extended ALGOL language which is similar to the ALGOL 60 language. Certain aspects of the simulation program are found in the following paragraphs.

a. Measures of Effectiveness: The simulation program prints out for each phase the following items: the number of repairmen, the sum of the service times, the sum of the queueing times, the number of times an arrival does not queue, the maximum queueing time, the total time to service 1,000 arrivals, and the total cost using seven different values of idle-time cost. Using these results, some important measures of effectiveness can be found.

The fraction of time the operators are servicing looms is the first and is found by dividing the sum of their service times by the product of the number of repairmen and the total time to service 1,000 arrivals. The second is the mean machine idle-time for a phase and is calculated by dividing the sum of the queueing times and service times by the number of arrivals. The mean queueing time of those who wait is the third and is found by dividing the sum of the queueing times by the number of ar-

rivals that wait for service.

The measures of effectiveness and the total cost for each phase describe the economical and practical effects of the priority and first-come-first-served queue disciplines.

b. Costs: The total cost for each repair phase considers the cost of looms being idle and the cost of the repairmen. As an example, the total cost of the prepare phase with a queue discipline of first-come-first-served is given. The idle-time cost for the I^{th} arrival is given as

$$W[I] = Y \cdot CQP[I] \cdot (QTP[I]/60 + STP[I]/60),$$

where Y = the downtime cost,

CQP [I] = a constant which is given the value of two if I is a priority item or one if it is not,

QTP [I] = the queueing time of arrival I, and

STP [I] = the service time of arrival I.

The downtime cost is varied in the computer program from the value of one to 25 in increments of four. This allows the examination of the effects of a priority queue discipline as the cost of downtime becomes larger.

It has been assumed* that 25 percent of the arrivals are of the priority classification. Priority items are assumed** to have a downtime

---

* This assumption is an intelligent estimate of the percent of priority items for this system.

** This assumption insures that priority items are serviced before non-priority items.

cost of twice that of non-priority items. A priority classification is assigned randomly to the arrivals.

The total cost of a phase is equal to the sum of the idle-time costs plus the product of the hourly wage rate, the number of repairmen, and the elapsed time in hours.

c. Estimate of the Variance of Service Times*: The variances of the different service times were unknown and had to be estimated. Time study data gave different ranges of values for the service times, and a method which used these ranges was employed to give unbiased estimates of the variances. The essence of the procedure is given below.

Let x' and σ' represent the mean and standard deviation respectively of the normal distribution. The "prime" will always represent a parameter of a probability distribution. $\bar{x}$ denotes the sample average and will be used as an estimate of the mean of the normal distribution.

The range (R) of a subgroup of n observations is defined as the difference between the largest and smallest value. Define $\bar{R}$ as the average of the ranges of k subgroups, that is,

$$\bar{R} = \frac{R_1 + R_2 + \cdots + R_k}{k}$$

An unbiased estimate of σ' is $\bar{R}/d_2$, where d_2 is a value found such that $E(\bar{R}/d_2) = \sigma'$. Values of d_2 can be found in a book on statistical quality control.

---

* Raw data were not available for this study, otherwise $\bar{x}$ and s^2 would have been used.

Comments

One of the major errors in debugging the program was that a segment was greater than 4,093 syllables. This error was the result of not having the program broken down into enough blocks. One prime use of the block is to obtain automatic segmentation of a program in order to cope with the realities of finite computer memories. The program was finally written so that portions which were inactive for a period of time became separate blocks. This procedure allowed a greater amount of core memory to be available for data. For a better explanation of blocks in ALGOL and of programming, the reader is referred to Thurnau, Johnson, and Ham (14).

The first arrival times were generated using the procedure that was explained for generating arrival-time intervals. The arrivals to the second phase are the departure times of the first phase. Since 15 different sets of departure times from the first phase are possible, a selection of one of the 15 sets had to be made. The departure-time set of the optimal number of repairmen based on the minimum total cost was chosen. The same procedure was used to determine the arrival times for the third phase.

When the queue discipline chooses an arrival for service, the arrival is always repaired by the operator who has or will finish first. This method is commonly practiced in industry.

It is certainly possible that the size of the program could be reduced to allow a smaller computer to be used. Methods can be devised to reduce the amount of computer storage. One could store arrival and service times on magnetic tape or punched cards after their generation, and then recall these values when they are needed. The program is far from

ideal, but it is a workable program for a computer the size of the Burroughs B 5000.

The program also illustrates a subprogram for generating random numbers by the congruential multiplicative method. The subprogram allows a random number to be called simply by writing the value R(C). The use of the subprogram eliminates the need of writing a random number procedure each time that a random number is required in the program.

CHAPTER IV

RESULTS

Definition of Terms

The following terms are used in interpreting the results of this study and are defined below. All times are given in minutes.

Phase I - Prepare

Channels - the number of repairmen.

SSTP - sum of the service times for the prepare phase.

MQTP - maximum queueing time for the prepare phase.

NQTP - the number of arrivals that do not wait for service at the prepare phase.

SQTP - the sum of the queueing times at the prepare phase.

TIME - the last departure time in minutes from the prepare phase.

TCP(I) - the total cost of the prepare phase with no priority queue discipline and an idle-time cost of I.

TCPP(I) - the total cost of the prepare phase with a priority queue discipline and an idle-time cost of I.

Phase II - Tie

Channels - the number of repairmen.

SSTT - sum of the service times for the tie phase.

MQTT - maximum queueing time for the tie phase.

NQTT - number of arrivals that do not wait for service at the tie phase.

SQTT - the sum of the queueing times at the tie phase.

TIME - the last departure time in minutes from the tie phase.

TCT(I) - the total cost of the tie phase with no priority queue discipline and an idle-time cost of I.

TCTT(I) - the total cost of the tie phase with a priority queue discipline and an idle-time cost of I.

Phase III - Finish

Channels - the number of repairmen.

SSTF - sum of the service times for the finish phase.

MQTF - maximum queueing time for the finish phase.

NQTF - the number of arrivals that do not wait for service at the finish phase.

SQTF - the sum of the queueing times at the finish phase.

TIME - the last departure time in minutes from the finish phase.

TCF(I) - the total cost of the finish phase with no priority queue discipline and an idle-time cost of I.

TCFF(I) - the total cost of the finish phase with a priority queue discipline and an idle-time cost of I.

Discussion of Results

Table 1, Table 4, and Table 7 indicate that Phase I, the phase of removing the old warp beam and replacing it with a full warp beam, gives the same results for whichever queue discipline is used. These results are influenced by the fact that a minimum of three repairmen for this phase were considered. As a consequence, only 17 arrivals needed to wait for service and the priority queue discipline was the same as being one

of first-come-first-served.

The operation of tying the old warp to the new warp is Phase II, and its results are given by Table 2, Table 5, and Table 8. The maximum queueing time of an arrival is greater using the priority queue discipline for channels three and four. However, the sum of the queueing times is less for channel three and equal for channel four using the priority queue discipline. This results in a lower mean queueing time and a reduction in the mean machine idle-time for channel three. Other slight differences are noted in the mean machine idle-time and the mean queueing time for an arrival that has to wait for service.

Some cost differences exist in Phase II for channels three and four. The priority queue discipline gives a slightly lower cost than the queue discipline of first-come-first-served. The number of repairmen for this phase depends upon the cost of idle-time.

Phase III is the operation of finish and is the last phase. Table 3, Table 6, and Table 9 furnish the results for this phase. The results are similar to those of Phase II and indicate that the priority queue discipline is only slightly better than a first-come-first-served queue discipline. The optimum number of workers ranges from three to five depending upon the idle-time cost.

Table 10 gives a summary of the optimum cost data for the entire system. The minimum cost for each of the seven different idle-time costs was tabulated for each phase of service. The results for both the first-come-first-served and priority queue disciplines are given. The total cost using the priority queue discipline is slightly lower for all seven values of idle-time cost.

Table 1. Comparison of Queue Disciplines for Phase I

Phase I - No Priority

Channels	SSTP	MQTP	NQTP	SQTP	TIME
3	25627.1	28.2	983	144.9	38839.5
4	25627.1	13.8	997	32.4	38839.5
5	25627.1	9.1	998	15.3	38839.5
6	25627.1	1.6	999	1.6	38839.5
7	25627.1	0.0	1000	0.0	38839.5
8	25627.1	0.0	1000	0.0	38839.5

Phase I - Priority

Channels	SSTP	MQTP	NQTP	SQTP	TIME
3	25627.1	34.5	983	144.9	38839.5
4	25627.1	17.1	997	32.4	38839.5
5	25627.1	9.1	998	15.3	38839.5
6	25627.1	1.6	999	1.6	38839.5
7	25627.1	0.0	1000	0.0	38839.5
8	25627.1	0.0	1000	0.0	38839.5

Table 2. Comparison of Queue Disciplines for Phase II

Phase II - No Priority

Channels	SSTT	MQTT	NQTT	SQTT	TIME
3	65603.5	82.7	822	3941.8	38912.0
4	65603.5	43.7	956	698.9	38912.0
5	65603.5	28.8	991	126.2	38912.0
6	65603.5	21.4	997	34.0	38912.0
7	65603.5	2.7	998	5.2	38912.0
8	65603.5	0.0	1000	0.0	38912.0
9	65603.5	0.0	1000	0.0	38912.0
10	65603.5	0.0	1000	0.0	38912.0

Phase II - Priority

Channels	SSTT	MQTT	NQTT	SQTT	TIME
3	65603.5	172.6	821	3846.5	38912.0
4	65603.5	62.7	956	698.9	38912.0
5	65603.5	27.3	991	126.2	38912.0
6	65603.5	19.9	997	34.0	38912.0
7	65603.5	2.4	998	3.6	38912.0
8	65603.5	0.0	1000	0.0	38912.0
9	65603.5	0.0	1000	0.0	38912.0
10	65603.5	0.0	1000	0.0	38912.0

Table 3. Comparison of Queue Disciplines for Phase III

Phase III - No Priority

Channels	SSTF	MQTF	NQTF	SQTF	TIME
3	91173.8	216.6	518	26734.2	38974.9
4	91173.8	115.3	861	3453.8	38974.8
5	91173.8	62.6	973	542.7	38974.8
6	91173.8	14.0	994	38.3	38974.8
7	91173.8	0.0	1000	0.0	38974.8
8	91173.8	0.0	1000	0.0	38974.8
9	91173.8	0.0	1000	0.0	38974.8
10	91173.8	0.0	1000	0.0	38974.8

Phase III - Priority

Channels	SSTF	MQTF	NQTF	SQTF	TIME
3	91173.8	480.2	516	21870.2	38974.9
4	91173.8	151.3	867	3205.9	38974.8
5	91173.8	75.0	975	491.7	38974.8
6	91173.8	14.0	994	38.3	38974.8
7	91173.8	0.0	1000	0.0	38974.8
8	91173.8	0.0	1000	0.0	38974.8
9	91173.8	0.0	1000	0.0	38974.8
10	91173.8	0.0	1000	0.0	38974.8

Table 4. Cost Comparisons of Queue Disciplines for Phase I

Phase I - No Priority

Channels	TCP(1)	TCP(5)	TCP(9)	TCP(13)	TCP(17)	TCP(21)	TCP(25)
3	5766.82	7860.78	9954.74	12048.70	14142.66	16236.62	18330.58
4	7512.29	9597.03	11681.78	13766.52	15851.26	17936.00	20020.74
5	9259.79	11343.39	13426.99	15510.60	17594.20	19677.81	21761.41
6	11007.33	13090.02	15172.70	17255.39	19338.08	21420.76	23503.45
7	12755.08	14837.67	16920.25	19002.83	21085.41	23168.00	25250.58
8	14502.86	16585.44	18668.03	20750.61	22833.19	24915.77	26998.35

Phase I - Priority

Channels	TCPP(1)	TCPP(5)	TCPP(9)	TCPP(13)	TCPP(17)	TCPP(21)	TCPP(25)
3	5766.82	7860.78	9954.74	12048.70	14142.66	16236.62	18330.58
4	7512.29	9597.03	11681.78	13766.52	15851.26	17936.00	20020.74
5	9259.79	11343.39	13426.99	15510.60	17594.20	19677.81	21761.41
6	11007.33	13090.02	15172.70	17255.39	19338.08	21420.76	23503.45
7	12755.08	14837.67	16920.25	19002.83	21085.41	23168.00	25250.58
8	14502.86	16585.44	18668.03	20750.61	22833.19	24915.77	26998.35

Table 5. Cost Comparisons of Queue Disciplines for Phase II

Phase II - No Priority

Channels	TCT(1)	TCT(5)	TCT(9)	TCT(13)	TCT(17)	TCT(21)	TCT(25)
3	4620.05	10259.28	15898.51	21537.75	27176.98	32816.21	38455.45
4	5622.85	10993.01	16363.16	21733.31	27103.46	32473.61	37843.76
5	6681.44	12005.61	17329.78	22653.95	27978.12	33302.29	38626.46
6	7749.10	13063.62	18378.14	23692.65	29007.17	34321.69	39636.20
7	8818.70	14131.29	19443.88	24756.47	30069.06	35381.65	40694.24
8	9888.69	15200.94	20513.19	25825.44	31137.68	36449.93	41762.18
9	10958.77	16271.02	21583.27	26895.51	32207.76	37520.01	42832.26
10	12028.85	17341.10	22653.35	27965.59	33277.84	38590.09	43902.33

Phase II - Priority

Channels	TCTT(1)	TCTT(5)	TCTT(9)	TCTT(13)	TCTT(17)	TCTT(21)	TCTT(25)
3	4615.22	10235.14	15855.06	21474.98	27094.90	32714.82	38334.74
4	5622.71	10992.27	16361.83	21731.40	27100.96	32470.53	37840.09
5	6681.44	12005.61	17329.78	22653.95	27978.12	33302.29	38626.46
6	7749.10	13063.62	18378.14	23692.65	29007.17	34321.69	39636.20
7	8818.67	14131.16	19443.64	24756.13	30068.61	35381.10	40693.58
8	9888.69	15200.94	20513.19	25825.44	31137.68	36449.93	41762.18
9	10958.77	16271.02	21583.27	26895.51	32207.76	37520.01	42832.26
10	12028.85	17341.10	22653.35	27965.59	33277.84	38590.09	43902.33

Table 6. Cost Comparisons of Queue Disciplines for Phase III

Phase III - No Priority

Channels	TCF(1)	TCF(5)	TCF(9)	TCF(13)	TCF(17)	TCF(21)	TCF(25)
3	5619.61	15236.33	24853.06	34469.78	44086.51	53703.24	63319.96
4	6220.53	13953.74	21686.94	29420.15	37153.35	44886.56	52619.77
5	7236.05	14744.11	22252.16	29760.22	37268.28	44776.34	52284.40
6	8297.82	15765.72	23233.61	30701.51	38169.41	45637.31	53105.21
7	9368.98	16834.31	24299.64	31764.97	39230.30	46695.63	54160.96
8	10440.79	17906.12	25371.45	32836.78	40302.10	47767.43	55232.76
9	11512.60	18977.92	26443.25	33908.58	41373.91	48839.24	56304.57
10	12584.40	20049.73	27515.06	34980.39	42445.72	49911.05	57376.38

Phase III - Priority

Channels	TCFF(1)	TCFF(5)	TCFF(9)	TCFF(13)	TCFF(17)	TCFF(21)	TCFF(25)
3	5497.95	14628.04	23758.13	32888.22	42018.31	51148.40	60278.49
4	6215.83	13930.22	21644.61	29358.99	37073.38	44787.77	52502.16
5	7235.09	14739.29	22243.49	29747.69	37251.89	44756.09	52260.29
6	8297.82	15765.72	23233.61	30701.51	38169.41	45637.31	53105.21
7	9368.98	16834.31	24299.64	31764.97	39230.30	46695.63	54160.96
8	10440.79	17906.12	25371.45	32836.78	40302.10	47767.43	55232.76
9	11512.60	18977.92	26443.25	33908.58	41373.91	48839.24	56304.57
10	12584.40	20049.73	27515.06	34980.39	42445.72	49911.05	57376.38

Table 7. Measures of Effectiveness for Phase I

Phase I - No Priority

Channels	Operator Utilization	Mean Machine Idle-time	Mean Queueing Time
3	22%	25.77	8.5
4	17%	25.66	10.8
5	13%	25.64	7.7
6	11%	25.63	1.6
7	9%	25.63	0.0
8	8%	25.63	0.0

Phase I - Priority

Channels	Operator Utilization	Mean Machine Idle-time	Mean Queueing Time
3	22%	25.77	8.5
4	17%	25.66	10.8
5	13%	25.64	7.7
6	11%	25.63	1.6
7	9%	25.63	0.0
8	8%	25.63	0.0

Table 8. Measures of Effectiveness for Phase II

Phase II - No Priority

Channels	Operator Utilization	Mean Machine Idle-time	Mean Queueing Time
3	56%	69.55	22.1
4	42%	66.30	15.9
5	34%	65.73	14.0
6	28%	65.64	11.3
7	24%	65.61	2.6
8	21%	65.60	0.0
9	19%	65.60	0.0
10	17%	65.60	0.0

Phase II - Priority

Channels	Operator Utilization	Mean Machine Idle-time	Mean Queueing Time
3	56%	69.45	21.5
4	42%	66.30	15.9
5	34%	65.73	14.0
6	28%	65.64	11.3
7	24%	65.61	1.8
8	21%	65.60	0.0
9	19%	65.60	0.0
10	17%	65.60	0.0

Table 9. Measures of Effectiveness for Phase III

Phase III - No Priority

Channels	Operator Utilization	Mean Machine Idle-time	Mean Queueing Time
3	78%	117.9	55.5
4	59%	94.6	24.8
5	47%	91.7	20.1
6	39%	91.2	6.4
7	33%	0.0	0.0
8	29%	0.0	0.0
9	26%	0.0	0.0
10	23%	0.0	0.0

Phase III - Priority

Channels	Operator Utilization	Mean Machine Idle-time	Mean Queueing Time
3	78%	113.0	45.2
4	59%	94.4	24.1
5	47%	91.7	19.7
6	39%	91.2	6.4
7	33%	0.0	0.0
8	29%	0.0	0.0
9	26%	0.0	0.0
10	23%	0.0	0.0

Table 10. Summary of Optimum Cost Data for the Entire System

No Priority

	I = 1	I = 5	I = 9	I = 13	I = 17	I = 21	I = 25
TCP(I)	$ 5766.82	$ 7860.78	$ 9954.74	$12048.70	$14142.66	$16236.62	$ 18330.58
TCT(I)	4620.05	10259.28	15898.51	21537.75	27103.46	32473.61	37843.76
TCF(I)	5619.61	13953.74	21686.94	29420.15	37153.35	44776.34	52284.40
TOTAL	$16006.48	$32073.80	$47540.19	$63006.60	$78399.47	$93486.57	$108458.74

Priority

	I = 1	I = 5	I = 9	I = 13	I = 17	I = 21	I = 25
TCPP(I)	$ 5766.82	$ 7860.78	$ 9954.74	$12048.70	$14142.66	$16236.62	$ 18330.58
TCTT(I)	4615.22	10235.14	15855.06	21474.98	27094.90	32470.53	37840.09
TCFF(I)	5497.95	13930.22	21644.61	29358.99	37073.38	44756.09	52260.29
TOTAL	$15879.99	$32026.14	$47454.41	$62882.67	$78310.94	$93463.24	$108430.96

CHAPTER V

CONCLUSIONS AND RECOMMENDATIONS

Conclusions

1. For this study, the priority queue discipline of the mean service time divided by its cost of queueing for a unit of time is only slightly more economical to use than a queue discipline of first-come-first-served.

2. The cost of implementing a priority queue discipline in the particular textile weaving process under study would not offset the savings gained.

3. The cost of a loom being idle is directly proportional to the optimum number of repairmen that should be employed at each phase for both types of queue disciplines.

4. The mean machine idle-time and the mean queueing time for an arrival that has to wait are lower when using the priority queue discipline.

5. The priority queue discipline tends to have a higher maximum queueing time than a first-come-first-served queue discipline.

6. The two queue disciplines compared in this study had the same rate of operator utilization for similar number of repairmen at each phase.

Recommendations

The simulation program consumes approximately 45 minutes of computer time using the Burroughs B 5000 computer. This period of time is fairly long considering the speed of the B 5000, but a more thorough examination of the general problem of economical comparisons of priority and first-come-first-served queue disciplines could be made by changing certain aspects of the program. One major change would be to increase the arrival rate. This change would more dramatically point out the influence of the priority queue discipline. Increasing the number of priority items and having more than two classifications of priority would better emphasize the differences in the two queue disciplines.

The simulation program is a workable program, but certain refinements could be made. The printing out of the results often duplicates previous data that were printed. However, this problem is not serious when the print out speed of the B 5000 computer is taken into consideration.

The most time-consuming part of the program is the repetition of calculation for the different number of channels. If the number of channels studied is kept to a minimum then the time of running the program is greatly reduced.

The applications of the program are numerous. One may choose different arrival-time and service-time distributions and still use the basic logic of the program. It is felt that once certain obstacles in logic are overcome in writing the program, further examinations of similar problems or further elaboration on the problem in this study may be made easily. The simulation program of this study should reduce the time

needed in examining more elaborate queueing processes which involve queues in series, multiple service times, and different queue disciplines.

Method of Attack for the Second Queueing Process

The second queueing process is that part of the system which is responsible for the maintenance of the looms while they are running. Three types of loom stoppage can occur, and these stoppages are repaired by the weavers.

The three classifications of weaver repair are warp breaks, filling breaks, and mechanical breaks. The arrival rate of these stoppages is 1,447 per eight hour shift with a range of 1,200 to 1,700 per eight hour shift. Warp breaks occur approximately 42 percent of the time, while filling breaks occur 55 percent of the time. The remaining three percent is of the mechanical type. The mean service rate is 1.25 minutes for warp breaks, 0.50 minute for filling breaks, and 0.30 minute for mechanical breaks. The arrivals are random, and the service times follow a normal distribution. The ranges for the service times can be found from time study data.

A simulation program for this queueing process would be similar to that of either of the last two repair phases considered in the simulation of queues in series. Adjustments would have to be made for the generation of arrival and service times. However, the large number of arrivals per hour and the short repair times would make it difficult to implement a priority queue discipline. In some textile mills the weavers attend the machines in a cyclic fashion and do not use either a priority or first-come-first-served queue discipline. An advantage of this method is that a closer supervision of the weavers can be accomplished.

APPENDIX A

```
          BEGIN
INTEGER       C,J,II,K,M,NQTP,NQTT,NQTF,L,I,ISAVE,NI,S,KK,V,N,RN,P,Q,

              F,FSAVE,JI,KI;
REAL          Z,Y,SSTP,SSTT,SSTF,SR,T,SQTP,SQTT,SQTF,MTP,MTT,MTF,
              MQTP,MQTT,MQTF,TIME,MPRP,MPRT,MPRF;
ARRAY         AT[0:1001], STP[0:1000], STT[0:1000], STF[0:1000],
              CQP[0:1000], TP[0:20], TT[0:25], TF[0:25], QTP[0:1000],
              QTT[0:1000], QTF[0:1000], DPT[0:1000], COST[0:25,0:25],
              W[0:1000], TCP[0:25], TCT[0:25], TCF[0:25],
              DPTP[0:1000], DPTT[0:1000], PIP[0:50],PRT[0:200],
              PRF[0:200],
              PRF[0:200],
              TCPP[0:25], TCTT[0:25], TCFF[0:25],
              DPTPP[0:1001], DPTTT[0:1001], B[0:1001] ;
FILE OUT      PRINT 1(2,15);
FILE OUT      PUNCH 2(2,15,SAVE 1 ) ;
LABEL         CALC1,CALC2,P1,P4,P5;
FORMAT OUT    FM1(X10,"CHANNELS",X6,"SSTP",X5,"MQTP",X5,"NQTP",X7,
```

```
                "SQTP",X8,"TIME")/

FORMAT OUT      FM2(X13,I2,X5,F8.1,X4,F5.1,X5,I4,X4,F7.1,X4,F8.1)/

FORMAT OUT      FM3(X1,"CHANNELS",X2,"TCP(1)",X4,"TCP(5)",X4,"TCP(9)",

                X4,"TCP(13)",X4,"TCP(17)",X4,"TCP(21)",X3,"TCP(25)")/

FORMAT OUT      FM4(X4,I2,X2,F9.2,X1,F9.2,X1,F9.2,X2,F9.2,X2,F9.2,

                X2,F9.2,X1,F9.2)/

FORMAT OUT      FM5(X1,"CHANNELS",X2,"TCPP(1)",X3,"TCPP(5)",X3,"TCPP(9)",

                X3,"TCPP(13)",X3,"TCPP(17)",X2,"TCPP(21)",X2,"TCPP(25)")/
FORMAT OUT      FM6(X4,I2,X3,F9.2,X1,F9.2,X1,F9.2,X2,F9.2,X2,F9.2,

                X1,F9.2,X1,F9.2)/

FORMAT OUT      FM7(X10,"CHANNELS",X6,"SSTT",X5,"MQTT",X5,"NQTT",X7,

                "SQTT",X8,"TIME")/

FORMAT OUT      FM8(X13,I2,X5,F8.1,X4,F5.1,X5,I4,X4,F7.1,X4,F8.1)/

FORMAT OUT      FM9(X1,"CHANNELS",X2,"TCT(1)",X4,"TCT(5)",X4,"TCT(9)",

                X4,"TCT(13)",X4,"TCT(17)",X4,"TCT(21)",X3,"TCT(25)")/
```

```
FORMAT OUT      FM10(X4,I2,X2,F9.2,X1,F9.2,X1,F9.2,X2,F9.2,X2,F9.2,
                X2,F9.2,X1,F9.2);

FORMAT OUT      FM11(X1,"CHANNELS",X2,"TCTT(1)",X3,"TCTT(5)",X3,
                "TCTT(9)",X3,"TCTT(13)",X3,"TCTT(17)",X2,"TCTT(21)",
                X2,"TCTT(25)");

FORMAT OUT      FM12(X4,I2,X3,F9.2,X1,F9.2,X1,F9.2,X2,F9.2,X2,F9.2,
                X1,F9.2,X1,F9.2);

FORMAT OUT      FM13(X10,"CHANNELS",X6,"SSTF",X5,"MQTF",X5,"NQTF",X7,
                "SQTF",X8,"TIME");

FORMAT OUT      FM14(X13,I2,X5,F8.1,X4,F5.1,X5,I4,X4,F7.1,X4,F8.1);


FORMAT OUT      FM15(X1,"CHANNELS",X2,"TCF(1)",X4,"TCF(5)",X4,"TCF(9)",
                X4,"TCF(13)",X4,"TCF(17)",X4,"TCF(21)",X3,"TCF(25)");

FORMAT OUT      FM16(X4,I2,X2,F9.2,X1,F9.2,X1,F9.2,X2,F9.2,X2,F9.2,
                X2,F9.2,X1,F9.2);

FORMAT OUT      FM17(X1,"CHANNELS",X2,"TCFF(1)",X3,"TCFF(5)",X3,
                "TCFF(9)",X3,"TCFF(13)",X3,"TCFF(17)",X2,"TCFF(21)",
                X2,"TCFF(25)");
```

```
FORMAT OUT      FM18(X4,I2,X3,F9.2,X1,F9.2,X1,F9.2,X2,F9.2,X2,F9.2,

                X1,F9.2,X1,F9.2)};


        REAL PROCEDURE R(C)    ;
                INTEGER C  ;
          BEGIN
                REAL RH, RL, CH, CL, P, YY;

                YY ← Z/3.14159 ;
                IF N < 1 THEN
           DOUBLE(37295981245.0,4199736829.0, ←, RH, RL)   ELSE
                DOUBLE (YY,Z,←,RH,RL) ;
           DOUBLE(C, C, ←, CH, CL)   ;
                DOUBLE(RH,RL,CH,CL,×,←,RH,RL)        ;
             P  :=  RL.[11:37]       ;
           P  :=  P × 1.0@-10           ;
            IF P > 10.0 THEN P :=  P - 10.0      ;
           IF P     > 1.0 THEN P :=  P× 1.0@-01   ;
           P  :=  P  ;
                END R  ;

                C ← 30517578125 ;
                TCP[5] ← 0.0 ;
                Z ← R(C)} AT[1] ← - (1.0/0.024167) × LN(Z)};
                N := N + 1   ;
                FOR J ← 2 STEP 1 UNTIL 1000 DO
          BEGIN Z ← R(C)} Y ← -(1.0/0.024167) × LN(Z)} AT[J] ← AT[J-1]+Y
          END} AT[1001] ← 9999999.9} SSTP ← 0.0;
                FOR J ← 1 STEP 1 UNTIL 1000 DO
          BEGIN SR ← 0.0} FOR II ← 1 STEP 1 UNTIL 10 DO
```

```
          BEGIN Z ← R(C)⁞ SR ← SR + Z
          END⁞ T ← ((SR - 5.0) / SQRT (10.0 / 12.0)) × (1.7808) + 26.0⁞
               STP[J] ← T⁞ SSTP ← SSTP + STP[J]
          END⁞ FOR J ← 1 STEP 1 UNTIL 1000 DO
          BEGIN Z ← R(C)⁞
               IF Z ≤ 0.25000000000 THEN CQP[J] ← 2.0 ELSE CQP[J] ← 1.0
          END⁞ WRITE (PRINT,FM1)⁞ FOR K ← 3 STEP 1 UNTIL 8  DO
          BEGIN FOR M ← 1 STEP 1 UNTIL K DO
          BEGIN TP[M] ← AT[M] + STP[M]⁞ QTP[M] ← 0.0
          END⁞ NQTP ← K⁞ SQTP ← 0.0⁞
               FOR L ← K+1 STEP 1 UNTIL 1000 DO
          BEGIN MTP ← TP[1]⁞ FOR I ← 1 STEP 1 UNTIL (K-1) DO
          BEGIN IF TP[I+1] < MTP THEN MTP ← TP[I+1]⁞
          END⁞ T ← 0⁞ ISAVE ← 1⁞
               FOR I ← I + 1 WHILE TP[I] > MTP DO ISAVE ← I+1⁞
               IF AT[L] ≥ TP[ISAVE] THEN GO TO CALC1
               ELSE GO TO CALC2⁞
CALC1⁞         NI ← L-K⁞ QTP[L] ← 0.0⁞ NQTP ← NQTP + 1⁞
               SQTP ← SQTP + QTP[L]⁞ DPT[NI] ← TP[ISAVE]⁞
               TP[ISAVE] ← AT[L] + STP[L]⁞ GO TO P1⁞
CALC2⁞         NI ← L-K⁞ QTP[L] ← TP[ISAVE] - AT[L]⁞
               SQTP ← SQTP + QTP[L]⁞ DPT[NI] ← TP[ISAVE]⁞
               TP[ISAVE] ← AT[L] + STP[L] + QTP[L]⁞ GO TO P1⁞
P1⁞       END⁞ MQTP ←  QTP[1]⁞ FOR I ← 1 STEP 1 UNTIL 999 DO
          BEGIN IF QTP[I+1] > MQTP THEN MQTP ← QTP[I+1]⁞
          END⁞ FOR II ← 1000-K+1 STEP 1 UNTIL 1000 DO
          BEGIN MTP ← TP[1]⁞ FOR NI ← 1 STEP 1 UNTIL K-1 DO
          BEGIN IF TP[NI+1] < MTP
               THEN MTP ← TP[NI+1]⁞
          END⁞ T ← 0⁞ ISAVE ← 1⁞ FOR I ← I+1 WHILE TP[I] > MTP DO
               ISAVE ← I+1⁞ DPT[II] ← TP[ISAVE]⁞ TP[ISAVE] ← 999999999.9
          END⁞ TIME ← DPT[1000]⁞
```

```
          FOR Y ← 1 STEP 4 UNTIL 25 DO
     BEGIN W[1] ← (Y) × (CQP[1]) × ((QTP[1]/60) +(STP[1]/60));
          FOR I ← 2 STEP 1 UNTIL 1000 DO
          W[I] ← (Y) × (CQP[I]) × ((QTP[I]/60) +(STP[I]/60)) +
                 W[I-1];
          TCP[Y] ← W[1000] + 2.70 × K × (TIME/60);
          COST[K,5] ← TCP[5];
          COST[2,5] ← 999999999.9;
          IF COST[K,5] < COST[K-1,5] THEN GO TO P4
          ELSE GO TO P5;
P4;       FOR S ← 1 STEP 1 UNTIL 1000 DO
          OPTP[S] ← OPT[S]; GO TO P5;
P5;       WRITE(PRINT,FM2,K,SSTP,MQTP,NQTP,SQTP,TIME); WRITE(PUNCH,
          FM2,K,SSTP,MQTP,NQTP,SQTP,TIME); WRITE(PRINT,FM4,K,
          TCP[1],TCP[5],TCP[9],TCP[13],TCP[17],TCP[21],TCP[25]);
          WRITE(PUNCH,FM4,K,TCP[1],TCP[5],TCP[9],TCP[13],TCP[17],
          TCP[21],TCP[25]);
     END;
     END; WRITE(PUNCH,FM1); WRITE(PUNCH,FM3);
     BEGIN LABEL  P62,P66,P67,P68,P70,P72,CALC7,CALC8,CALC8A,
                  CALC8B;
          TCPP[5] ← 0.0 ;
          FOR S ← 1 STEP 1 UNTIL 1000 DO
          B[S] ← AT[S] ;
               WRITE (PRINT,FM1); FOR K ← 3 STEP 1 UNTIL 8 DO
     BEGIN FOR M ← 1 STEP 1 UNTIL K DO
     BEGIN TP[M] ← AT[M] + STP[M]; QTP[M] ← 0.0
     END; NQTP ← K; SQTP ← 0.0; KK ← 0;
P62;      MTP ← TP[1]; FOR I ← 1 STEP 1 UNTIL K-1 DO
     BEGIN IF TP[I+1] < MTP THEN MTP ← TP[I+1];
     END; T ← 0; ISAVE ← 1;
```

```
                FOR I ← I+1 WHILE TP[I] > MTP DO ISAVE ← I+1;
                L ← K; L ← L+1;
P66:            IF AT[L] = 999999.9 THEN
           BEGIN L ← L+1; GO TO P66
           END  ELSE GO TO P67;
P67:            IF AT[L] = 9999999.9 THEN GO TO P68
                ELSE IF AT[L] ≥ TP[ISAVE] THEN GO TO CALC7
                ELSE GO TO CALC8;
CALC7:          QTP[L] ← 0.0; NQTP ← NQTP + 1; SQTP ← SQTP + QTP[L];
                DPT[KK+1] ← TP[ISAVE];
                KK ← KK + 1  ;
                TP[ISAVE] ← AT[L] + STP[L];
                AT[L] ← 999999.9 ; GO TO P62 ;
CALC8:          FOR V ← 2 STEP 1 UNTIL 50 DO PRP[V] ← 999999999.9;
                N ← L; NI ← 1;
CALC8A:         IF (AT[N] <TP[ISAVE]) AND (AT[N] ≠ 999999.9) THEN
           BEGIN PRP[NI] ← STP[N] / CQP[N]; N ← N+1; NI ← NI+1;
                GO TO CALC8A
           END  ELSE IF AT[N] = 999999.9 THEN
           BEGIN N ← N + 1 ;  NI ← NI + 1 ;  GO TO CALC8A
           END  ELSE GO TO CALC8B;
CALC8B:         MPRP ← PRP[1]; FOR F ← 1 STEP 1 UNTIL 49 DO
           BEGIN IF PRP[F+1] < MPRP THEN MPRP ← PRP[F+1];
           END; F ← 0; FSAVE ← 1; FOR F ← F+1 WHILE PRP[F] > MPRP DO
                FSAVE ← F+1;
                L ← L + FSAVE - 1; QTP[L] ← TP[ISAVE] - AT[L];
                SQTP ← SQTP + QTP[L]; DPT[KK+1] ← TP[ISAVE];
                KK ← KK + 1  ;
                TP[ISAVE] ← AT[L] + STP[L] + QTP[L];
                AT[L] ← 999999.9 ; GO TO P62 ;
P68:            MQTP ← QTP[1]; FOR I ← 1 STEP 1 UNTIL 999 DO
           BEGIN IF QTP[I+1] > MQTP THEN MQTP ← QTP[I+1];
```

```
          END; FOR II ← 1000-K+1 STEP 1 UNTIL 1000 DO
          BEGIN MTP ← TP[1]; FOR NI ← 1 STEP 1 UNTIL K-1 DO
          BEGIN IF TP[NI+1] < MTP
               THEN MTP ← TP[NI+1];
          END; I ← 0; ISAVE ← 1; FOR I ← I+1 WHILE TP[I] > MTP DO
               ISAVE ← I+1; DPT[II] ← TP[ISAVE]; TP[ISAVE] ← 999999999.9
          END; TIME ← DPT[1000];
               FOR Y ← 1 STEP 4 UNTIL 25 DO
          BEGIN W[1] ← (Y) × (CQP[1]) × ((QTP[1]/60) +(STP[1]/60));
               FOR I ← 2 STEP 1 UNTIL 1000 DO
               W[I] ← (Y) × (CQP[I]) × ((QTP[I]/60) + (STP[I]/60)) +
                      W[I-1];
               TCPP[Y] ← W[1000] + 2.70 × K × (TIME/60);
               COST[K,5] ← TCPP[5];
               COST[2,5] ← 999999999.9;
               IF COST[K,5] < COST[K-1,5] THEN GO TO P70
               ELSE GO TO P72;
P70;           FOR S ← 1 STEP 1 UNTIL 1000 DO
               DPTPP[S] ← DPT[S]; DPTPP[1001] ← 9999999.9; GO TO P72;
P72;           WRITE(PRINT,FM2,K,SSTP,MQTP,NQTP,SQTP,TIME); WRITE(PUNCH,
               FM2,K,SSTP,MQTP,NQTP,SQTP,TIME); WRITE(PRINT,FM6,K,
               TCPP[1],TCPP[5],TCPP[9],TCPP[13],TCPP[17],TCPP[21],
               TCPP[25]); WRITE(PUNCH,FM6,K,TCPP[1],TCPP[5],TCPP[9],
               TCPP[13],TCPP[17],TCPP[21],TCPP[25]);
          END;
               FOR S ← 1 STEP 1 UNTIL 1000 DO
               AT[S] ← B[S] ;
          END; WRITE(PUNCH,FM5);
               FOR II ← 1 STEP 1 UNTIL 1001 DO
          BEGIN MPRP ← DPTPP[1];  FOR I ← 1 STEP 1 UNTIL 1000 DO
          BEGIN IF DPTPP[I+1] < MPRP THEN MPRP ← DPTPP[I+1] ;
          END; F ← 0 ; FSAVE ← 1 ;
```

```
        FOR F ← F + 1 WHILE DPTPP[F] > MPRP DO
        FSAVE ← F + 1 ;
        B[I] ← DPTPP[FSAVE] ;
        DPTPP[FSAVE] ← 99999999.9 ;
    END; FOR S ← 1 STEP 1 UNTIL 1001 DO
        DPTPP[S] ← B[S] ;
    END;

BREAK ;
    BEGIN LABEL  T1,T2,T4,T5,CALC3,CALC4;

        TCT[5] ← 0.0 ;
        SSTT ← 0.0; FOR J ← 1 STEP 1 UNTIL 1000 DO
    BEGIN B[J] ← R(C);
        Z ← B[J] ;
        IF B[J] < 0.500000000 THEN
    BEGIN
        SR ← 0.0; FOR II ← 1 STEP 1 UNTIL 10 DO
    BEGIN Z ← R(C); SR ← SR + Z;
    END; T ← ((SR -5.0)/SQRT (10.0/12.0)) × (4.5792) + 56.7;
        STT[J] ← T; SSTT ← SSTT + STT[ J];
    END
        ELSE IF B[J] ≥ 0.750000000 THEN
    BEGIN
        SR ← 0.0; FOR JI ← 1 STEP 1 UNTIL 10 DO
    BEGIN Z ← R(C); SR ← SR + Z;
    END; T ← ((SR -5.0)/SQRT (10.0/12.0)) × 17.8080 + 78.9;
        STT[J] ← T; SSTT ← SSTT + STT[J];
    END  ELSE
    BEGIN SR ← 0.0; FOR KI ← 1 STEP 1 UNTIL 10 DO
    BEGIN Z ← R(C); SR ← SR +Z;
    END; T ← ((SR -5.0)/SQRT (10.0/12.0)) × 22.1328 + 85.7;
```

```
                STT[ J] ← T; SSTT ← SSTT + STT[J] ;
           END  ;
T1:        END; WRITE (PRINT,FM7) ;    FOR K ← 3 STEP 1 UNTIL 10 DO
           BEGIN FOR M ← 1 STEP 1 UNTIL K DO
           BEGIN TT[M] ← DPTP[M] + STT[M]; QTT[M] ← 0.0
           END; NQTT ← K; SQTT ← 0.0;
                FOR L ← K+1 STEP 1 UNTIL 1000 DO
           BEGIN MTT ← TT[1]; FOR I ← 1 STEP 1 UNTIL (K-1) DO
           BEGIN IF TT[I+1] < MTT THEN MTT ← TT[I+1];
           END; I ← 0; ISAVE ← 1;
                FOR I ← I + 1 WHILE TT[I] > MTT DO ISAVE ← I+1;
                IF DPTP[L] ≥ TT[ISAVE] THEN GO TO CALC3
                ELSE GO TO CALC4;
CALC3:          NI ← L-K; QTT[L] ← 0.0; NQTT ← NQTT + 1;
                SQTT ← SQTT + QTT[L]; DPT[NI] ← TT[ISAVE];
                TT[ISAVE] ← DPTP[L] + STT[L]; GO TO T2;
CALC4:          NI ← L-K; QTT[L] ← TT[ISAVE] - DPTP[L];
                SQTT ← SQTT + QTT[L]; DPT[NI] ← TT[ISAVE];
                TT[ISAVE] ← DPTP[L] + STT[L] + QTT[L]; GO TO T2;
T2:        END; MQTT ← QTT[1]; FOR I ← 1 STEP 1 UNTIL 999 DO
           BEGIN IF QTT[I+1] > MQTT THEN MQTT ← QTT[I+1];
           END; FOR II ← 1000 -K+1 STEP 1 UNTIL 1000 DO
           BEGIN MTT ← TT[1]; FOR NI ← 1 STEP 1 UNTIL K-1 DO
           BEGIN IF TT[NI+1] < MTT
                THEN MTT ← TT[NI+1];
           END; I ← 0; ISAVE ← 1; FOR I ← I+1 WHILE TT[I] > MTT DO
                ISAVE ← I+1;
                DPT[II] ← TT[ISAVE]; TT[ISAVE] ← 999999999.9
           END; TIME ← DPT[1000];
                FOR Y ← 1 STEP 4 UNTIL 25 DO
           BEGIN W[1] ← (Y) × (COP[1]) × ((QTT[1]/60) + (STT[1]/60));
                FOR I ← 2 STEP 1 UNTIL 1000 DO
```

```
                W[I] ← (Y) × (CQP[I]) × ((QTT[I]/60) + (STT[I]/60)) +
                       W[Y-1];
                TCT[Y] ← W[1000] + 1.65 × K × (TIME/60);
                COST[K,5] ← TCT[5];
                COST[2,5] ← 999999999.9;
                IF COST[K,5] < COST[K-1,5] THEN GO TO T4
                ELSE GO TO T5;
T4:             FOR S ← 1 STEP 1 UNTIL 1000 DO
                DPTT[S] ← DPT[S]; GO TO T5;
T5:             WRITE(PRINT,FM8,K,SSTT,MQTT,NQTT,SQTT,TIME); WRITE(PUNCH,
                FM8,K,SSTT,MQTT,NQTT,SQTT,TIME); WRITE(PRINT,FM10,K,
                TCT[1],TCT[5],TCT[9],TCT[13],TCT[17],TCT[21],TCT[25]);
                WRITE(PUNCH,FM10,K,TCT[1],TCT[5],TCT[9],TCT[13],TCT[17],
                TCT[21],TCT[25]);
            END;
            END; WRITE(PUNCH,FM7); WRITE(PUNCH,FM9);
                FOR II ← 1 STEP 1 UNTIL 1000 DO
            BEGIN MPRT ← DPTT[1] ; FOR I ← 1 STEP 1 UNTIL   999  DO
            BEGIN IF DPTT[I+1] < MPRT THEN MPRT ← DPTT[I+1] ;
            END; F ← 0 ;  FSAVE ← 1 ;
                FOR F ← F + 1 WHILE DPTT[F] > MPRT DO
                FSAVE ←  F + 1 ;
                B[I] ← DPTT[FSAVE] ;
                DPTT[FSAVE] ← 99999999.9 ;
            END; FOR S ← 1 STEP 1 UNTIL 1000 DO
                DPTT[S] ← B[S] ;
            END;

  BREAK ;
            BEGIN LABEL T62,T66,T67,T68,T70,T72,CALC9,CALC10,
                        CALC10A,CALC10B;
```

```
                TCTT[5] ← 0.0 ;
                FOR S ← 1 STEP 1 UNTIL 1000 DO
                B[S] ← DPTPP[S] ;
                WRITE (PRINT,FM7) ;     FOR K ← 3 STEP 1 UNTIL 10 DO
          BEGIN FOR M ← 1 STEP 1 UNTIL K DO
          BEGIN TT[M] ← DPTPP[M] + STT[M]; QTT[M] ← 0.0;
          END; NQTT ← K; SQTT ← 0.0; KK ← 0;
T62:            MTT ← TT[1]; FOR I ← 1 STEP 1 UNTIL K-1 DO
          BEGIN IF TT[I+1] < MTT THEN MTT ← TT[I+1];
          END; I ← 0; ISAVE ← 1;
                FOR I ← I+1 WHILE TT[I] > MTT DO ISAVE ← I+1;
                L ← K; L ← L+1;
T64:            IF DPTPP[L] = 999999.9 THEN
          BEGIN L ← L+1; GO TO T66
          END  ELSE GO TO T67;
T67:            IF DPTPP[L] = 9999999.9 THEN GO TO T68
                ELSE IF DPTPP[L] ≥ TT[ISAVE] THEN GO TO CALC9
                ELSE GO TO CALC10;
CALC9:          QTT[L] ← 0.0; NQTT ← NQTT + 1; SQTT ← SQTT + QTT[L];
                DPT[KK+1] ← TT[ISAVE];
                KK ← KK + 1  ;
                TT[ISAVE] ← DPTPP[L] + STT[L];
                DPTPP[L] ← 999999.9 ; GO TO T62 ;
CALC10:         FOR V ← 2 STEP 1 UNTIL 186 DO PRT[V] ← 99999.9 ;
                N ← L; NI ← 1;
                DO BEGIN
CALC10A:        IF (DPTPP[N] < TT[ISAVE])AND (DPTPP[N] ≠ 999999.9) THEN
          BEGIN PRT[NI] ← STT[N]/CQP[N]; N ← N+1; NI ← NI+1;
                GO TO CALC10A
          END  ELSE IF DPTPP[N] = 999999.9 THEN
          BEGIN N ← N + 1 ;  NI ← NI + 1 ;  GO TO CALC10A
          END  ELSE GO TO CALC10B;
```

```
          END UNTIL NI = 186 ;
CALC10B;      MPRT ← PRT[1]; FOR F ← 1 STEP 1 UNTIL 185 DO
          BEGIN IF PRT[F+1] < MPRT THEN MPRT ← PRT[F+1];
          END; F ← 0; FSAVE ← 1; FOR F ← F+1 WHILE PRT[F] > MPRT DO
               FSAVE ← F+1;
               L ← L + FSAVE - 1; QTT[L] ← TT[ISAVE] - DPTPP[L];
               SQTT ← SQTT + QTT[L]; DPT[KK+1] ← TT[ISAVE];
               KK ← KK + 1  ;
               TT[ISAVE] ← DPTPP[L] + STT[L] + QTT[L];
               DPTPP[L] ← 999999.9 ; GO TO T62 ;
T6B;          MQTT ← QTT[1]; FOR I ← 1 STEP 1 UNTIL 999 DO
          BEGIN IF QTT[I+1] > MQTT THEN MQTT ← QTT[I+1];
          END; FOR II ← 1000-K+1 STEP 1 UNTIL 1000 DO
          BEGIN MTT ← TT[1]; FOR NI ← 1 STEP 1 UNTIL K-1 DO
          BEGIN IF TT[NI+1] < MTT
               THEN MTT ← TT[NI+1];
          END; T ← 0; ISAVE ← 1; FOR I ← I+1 WHILE TT[I] > MTT DO
               ISAVE ← I+1; DPT[II] ← TT[ISAVE]; TT[ISAVE] ← 999999999.9
          END; TIME ← DPT[1000];
               FOR Y ← 1 STEP 4 UNTIL 25 DO
          BEGIN W[1] ← (Y) × (CQP[1]) × ((QTT[1]/60) + (STT[1]/60));
               FOR I ← 2 STEP 1 UNTIL 1000 DO
               W[I] ← (Y) × (CQP[I]) × ((QTT[I]/60) + (STT[I]/60)) +
                      W[I-1];
               TCTT[Y] ← W[1000] + 1.65 × K × (TIME/60);
               COST[K,5] ← TCTT[5];
               COST[2,5] ← 999999999.9;
               IF COST[K,5] < COST[K-1,5] THEN GO TO T70
               ELSE GO TO T72;
T70;           FOR S ← 1 STEP 1 UNTIL 1000 DO
               DPTTT[S] ← DPT[S]; DPTTT[1001] ← 9999999.9; GO TO T72;
T72;           WRITE(PRINT,FMA,K,SSTT,MQTT,NQTT,SQTT,TIME); WRITE(PUNCH,
```

```
        FM8,K,SSTT,MQTT,NQTT,SQTT,TIME)} WRITE(PRINT,FM12,K,
        TCTT[1],TCTT[5],TCTT[9],TCTT[13],TCTT[17],TCTT[21],
        TCTT[25])} WRITE(PUNCH,FM12,K,TCTT[1],TCTT[5],TCTT[9],
        TCTT[13],TCTT[17],TCTT[21],TCTT[25])}
    END}
        FOR S ← 1 STEP 1 UNTIL 1000 DO
        OPTPP[S] ← B[S] }
    END} WRITE(PUNCH,FM11)}
        FOR II ← 1 STEP 1 UNTIL 1001 DO
    BEGIN MPRT ← OPTTT[1]}  FOR I ← 1 STEP 1 UNTIL 1000 DO
    BEGIN IF OPTTT[I+1] < MPRT THEN MPRT ← OPTTT[I+1] }
    END} F ← 0 } FSAVE ← 1 }
        FOR F ← F + 1 WHILE OPTTT[F] > MPRT DO
        FSAVE ← F + 1 }
        B[II] ← OPTTT[FSAVE] }
        OPTTT[FSAVE] ← 99999999.9 }
    END} FOR S ← 1 STEP 1 UNTIL 1001 DO
        OPTTT[S] ← B[S] }
    END}

BREAK }
    BEGIN LABEL F1,F2,CALC5,CALC6}

        SSTF ← 0.0} FOR J ← 1 STEP 1 UNTIL 1000 DO
    BEGIN B[J] ← R(C)}
        Z ← B[J] }
        IF B[J] < 0.650000000 THEN
    BEGIN
        SR ← 0.0} FOR II ← 1 STEP 1 UNTIL 10 DO
    BEGIN Z ← R(C)} SR ← SR + Z}
    END} T ← ((SR -5.0)/SQRT (10.0/12.0)) × 4.63008 + 56.7}
        STF[J] ← T} SSTF ← SSTF + STF[J]}
```

```
          END
               ELSE IF B[J] ≥ 0.750000000 THEN
          BEGIN
               SR ← 0.0; FOR JI ← 1 STEP 1 UNTIL 10 DO
          BEGIN Z ← R(C); SR ← SR + Z;
          END; T ← ((SR -5.0)/SQRT (10.0/12.0)) × 14.5008 + 210.0;
               STF[J] ← T; SSTF ← SSTF + STF[J];
          END  ELSE
          BEGIN SR ← 0.0; FOR KI ← 1 STEP 1 UNTIL 10 DO
          BEGIN Z ← R(C); SR ← SR + Z;
          END; T ← ((SR -5.0)/SQRT (10.0/12.0)) × 8.44608 + 93.4;
               STF[J] ← T; SSTF ← SSTF + STF[J];
          END;
F1:       END; WRITE(PRINT,FM13);    FOR K ← 3 STEP 1 UNTIL 15 DO
          BEGIN FOR M ← 1 STEP 1 UNTIL K DO
          BEGIN TF[M] ← DPTT[M] + STF[M]; QTF[M] ← 0.0
          END; NQTF ← K; SQTF ← 0.0;
               FOR L ← K+1 STEP 1 UNTIL 1000 DO
          BEGIN MTF ← TF[1]; FOR I ← 1 STEP 1 UNTIL (K-1) DO
          BEGIN IF TF[I+1] < MTF THEN MTF ← TF[I+1];
          END; T ← 0; ISAVE ← 1;
               FOR I ← I + 1 WHILE TF[I] > MTF DO ISAVE ← I+1;
               IF DPTT[L] ≥ TF[ISAVE] THEN GO TO CALC5
               ELSE GO TO CALC6;
CALC5:         NI ← L-K; QTF[L] ← 0.0; NQTF ← NQTF + 1;
               SQTF ← SQTF + QTF[L]; DPT[NI] ← TF[ISAVE];
               TF[ISAVE] ← DPTT[L] + STF[L]; GO TO F2;
CALC6:         NI ← L-K; QTF[L] ← TF[ISAVE] - DPTT[L];
               SQTF ← SQTF + QTF[L]; DPT[NI] ← TF[ISAVE];
               TF[ISAVE] ← DPTT[L] + STF[L] + QTF[L]; GO TO F2;
F2:       END; MQTF ← QTF[1]; FOR I ← 1 STEP 1 UNTIL 999 DO
          BEGIN IF QTF[I+1] > MQTF THEN MQTF ← QTF[I+1];
```

```
END; FOR II ← 1000 -K+1 STEP 1 UNTIL 1000 DO
BEGIN MTF ← TF[1]; FOR NI ← 1 STEP 1 UNTIL K-1 DO
BEGIN IF TF[NI+1] < MTF
     THEN MTF ← TF[NI+1];
END; I ← 0; ISAVE ← 1; FOR I ← I+1 WHILE TF[I] > MTF DO
     ISAVE ← I+1;
     DPT[II] ← TF[ISAVE]; TF[ISAVE] ← 999999999.9
END; TIME ← DPT[1000];
     FOR Y ← 1 STEP 4 UNTIL 25 DO
BEGIN W[1] ← (Y) × (CQP[1]) × ((QTF[1]/60) +(STF[1]/60));
     FOR I ← 2 STEP 1 UNTIL 1000 DO
     W[I] ← (Y) × (CQP[I]) × ((QTF[I]/60) +(STF[I]/60)) +
            W[I-1];
     TCF[Y] ← W[1000] + 1.65 × K × (TIME/60);
     WRITE(PRINT,FM14,K,SSTF,MQTF,NQTF,SQTF,TIME);
     WRITE(PUNCH,FM14,K,SSTF,MQTF,NQTF,SQTF,TIME);
     WRITE(PRINT,FM16,K,TCF[1],TCF[5],TCF[9],TCF[13],TCF[17],
     TCF[21],TCF[25]); WRITE(PUNCH,FM16,K,TCF[1],TCF[5],
     TCF[9],TCF[13],TCF[17],TCF[21],TCF[25]);
END;
END; WRITE(PUNCH,FM13); WRITE(PUNCH,FM15);
END;

BEGIN LABEL F62,F66,F67,F68,CALC11,CALC12,

            CALC12A,CALC12B;
     FOR S ← 1 STEP 1 UNTIL 1000 DO
     R[S] ← DPTTT[S] ;
     WRITE(PRINT,FM13);    FOR K ← 3 STEP 1 UNTIL 15 DO
BEGIN FOR M ← 1 STEP 1 UNTIL K DO
BEGIN TF[M] ← DPTTT[M] + STF[M]; QTF[M] ← 0.0;
END; NQTF ← K; SQTF ← 0.0; KK ← 0;
```

```
F62:          MTF ← TF[1]; FOR I ← 1 STEP 1 UNTIL K-1 DO
          BEGIN IF TF[I+1] < MTF THEN MTF ← TF[I+1];
          END; I ← 0; ISAVE ← 1;
               FOR I ← I+1 WHILE TF[I] > MTF DO ISAVE ← I+1;
               L ← K; L ← L+1;
F64:           IF DPTTT[L] = 999999.9 THEN
          BEGIN L ← L+1; GO TO F66
          END  ELSE GO TO F67;
F67:           IF DPTTT[L] = 9999999.9 THEN GO TO F68
               ELSE IF DPTTT[L] ≥ TF[ISAVE] THEN GO TO CALC11
               ELSE GO TO CALC12;
CALC11:        QTF[L] ← 0.0; NQTF ← NQTF + 1; SQTF ← SQTF + QTF[L];
               DPT[KK+1] ← TF[ISAVE];
               KK ← KK + 1  ;
               TF[ISAVE] ← DPTTT[L] + STF[L];
               DPTTT[L] ← 999999.9 ;  GO TO F62 ;
CALC12:        FOR V ← 2 STEP 1 UNTIL  186 DO PRF[V] ← 99999.9 ;
               N ← L; NI ← 1;
               DO BEGIN
CALC12A:       IF (DPTTT[N]< TF[ISAVE]) AND (DPTTT[N] ≠ 999999.9) THEN
          BEGIN PRF[NI] ← STF[N]/CQP[N]; N ← N+1; NI ← NI+1;
               GO TO CALC12A
          END  ELSE IF DPTTT[N] = 999999.9 THEN
          BEGIN N ← N + 1 ;  NI ← NI + 1 ;  GO TO CALC12A
          END  ELSE GO TO CALC12B;
          END UNTIL NI = 186 ;
CALC12B:       MPRF ← PRF[1]; FOR F ← 1 STEP 1 UNTIL  185 DO
          BEGIN IF PRF[F+1] < MPRF THEN MPRF ← PRF[F+1];
          END; F ← 0; FSAVE ← 1; FOR F ← F+1 WHILE PRF[F] > MPRF DO
               FSAVE ← F+1;
               C ← L + FSAVE - 1; QTF[L] ← TF[ISAVE] - DPTTT[L];
               SQTF ← SQTF + QTF[L]; DPT[KK+1] ← TF[ISAVE];
```

```
            KK ← KK + 1  ;
            TF[ISAVE] ← DPTTT[L] + STF[L] + QTF[L];
            DPTTT[L] ← 999999.9 ;  GO TO F62 ;
F68;        MQTF ← QTF[1]; FOR I ← 1 STEP 1 UNTIL 999 DO
          BEGIN IF QTF[I+1] > MQTF THEN MQTF ← QTF[I+1];
          END; FOR II ← 1000-K+1 STEP 1 UNTIL 1000 DO
          BEGIN MTF ← TF[1]; FOR NI ← 1 STEP 1 UNTIL K-1 DO
          BEGIN IF TF[NI+1] < MTF
            THEN MTF ← TF[NI+1];
          END; I ← 0; ISAVE ← 1; FOR I ← I+1 WHILE TF[I] > MTF DO
            ISAVE ← I+1; DPT[II] ← TF[ISAVE]; TF[ISAVE] ← 999999999.9
          END; TIME ← DPT[1000];
            FOR Y ← 1 STEP 4 UNTIL 25 DO
          BEGIN W[1] ← (Y) × (COP[1]) × ((QTF[1]/60) + (STF[1]/60));
            FOR I ← 2 STEP 1 UNTIL 1000 DO
            W[I] ← (Y) × (COP[I]) × ((QTF[I]/60) + (STF[I]/60)) +
                   W[I-1];
            TCFF[Y] ← W[1000] + 1.65 × K × (TIME/60);
            WRITE(PRINT,FM14,K,SSTF,MQTF,NQTF,SQTF,TIME);
            WRITE(PUNCH,FM14,K,SSTF,MQTF,NQTF,SQTF,TIME);
            WRITE(PRINT,FM18,K,TCFF[1],TCFF[5],TCFF[9],TCFF[13],
            TCFF[17],TCFF[21],TCFF[25]);
          END;
            FOR S ← 1 STEP 1 UNTIL 1000 DO
            DPTTT[S] ← B[S] ;
          END; WRITE(PUNCH,FM17);
          END;

          END.
```

BIBLIOGRAPHY

Literature Cited

1. P. J. Burke, "The Output of a Queueing System," Operations Research, vol. 4, pp. 699-704, 1956.

2. Burroughs Corporation, The B 5000 Concept, Bulletin 5000-20001-P, The Burroughs Corporation, Detroit, Michigan

3. A. Cobham, "Priority Assignment in Waiting Line Problems," Journal of Operations Research Society of America, vol. 2, pp. 70-76, 1954; "A Correction," ibid, vol. 3, pp. 547, 1955.

4. D. R. Cox and W. L. Smith, Queues, John Wiley and Sons, Inc., New York, 1961.

5. P. D. Finch, "The Output Process of the Queueing System M/G/1," Journal of Royal Statistical Society, Series B, vol. 21, no. 2, pp. 375-380, 1959.

6. J. L. Holley, "Waiting Line Subject to Priorities," Journal of Operations Research Society of America, vol. 2, pp. 341, 1954.

7. G. C. Hunt, "Sequential Arrays of Waiting Lines," Operations Research, vol. 4, no. 6, pp. 674-683, December, 1956.

8. J. R. Jackson, "Networks of Waiting Lines," Operations Research, vol. 5, no. 4, August, 1957.

9. R. R. P. Jackson, "Queueing Processes with Phase-type Service," Journal of Royal Statistical Society, Series B, vol. 18, no. 1, pp. 129-132, 1956.

10. H. Kesten and J. T. Runnenburg, "Some Elementary Proofs in Renewal Theory with Applications to Waiting Times," Mathematical Centrum Amsterdam, Statistical Afdeling Report S 203, pp. 16, 1956.

11. T. E. Phipps, Jr., "Machine Repair as a Priority Waiting-line Problem," Operations Research, vol. 4, pp. 76-85, 1956.

12. The RAND Corporation, A Million Random Digits, The Free Press, Glencoe, 1955.

13. E. Reich, "Waiting Times When Queues are in Tandem," Annuals of Mathematical Statistics, vol. 28, no. 3, pp. 768, September, 1957.

BIBLIOGRAPHY (Continued)

14. D. H. Thurman, R. E. Johnson, and R. J. Ham, ALGOL Programming, A Basic Approach, Big Mountain Press, 1964

Other References

15. Barry, J. Y., "A Priority Queueing Problem," Operations Research, vol. 4, pp. 385-386, 1956.

16. Benson, F., "Further Notes on the Productivity of Machines Requiring Attention at Random Intervals," Journal of Royal Statistical Society, Series B, vol. 14, pp. 200-210, 1952.

17. Benson, F. and D. R. Cox, "The Productivity of Machines Requiring Attention at Random Intervals," Journal of Royal Statistical Society, Series B, vol. 13, pp. 65-82, 1951.

18. Brigham, G., "On a Congestion Problem in an Aircraft Factory," Journal of Operations Research Society of America, vol. 3, pp. 412-428, 1955.

19. Goddard, L. S., Mathematical Techniques of Operational Research, Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1963.

20. Jackson, R. R. P., "Queueing Systems with Phase-type Service," Operations Research Quarterly, vol. 5, pp. 109-120, 1954.

21. Jackson, J. R., "Queues with Dynamic Priority Discipline," Management Science, vol. 8, no. 1, pp. 18-34, October, 1961.

22. Koenigsberg, E., "Finite Queues and Cyclic Queues," Operations Research, vol. 4, pp. 213-220, 1956.

23. Koenigsberg, E., "Cyclic Queues," Operational Research Quarterly, vol. 9, no. 1, 1958.

24. Morse, P. M., Queues, Inventories, and Maintenance, John Wiley and Sons, Inc., New York, 1963.

25. O'Brien, G. G., "The Solution of Some Queueing Problems," Journal of Society for Industrial and Applied Mathematics, vol. 2, pp. 133-149, 1954.

26. Rowe, A. J. and J. R. Jackson, "Research Problems in Production Scheduling," Journal of Industrial Engineering, May-June, 1956.

27. Saaty, T. L., Elements of Queueing Theory with Applications, McGraw-Hill Book Company, Inc., New York, 1961.

BIBLIOGRAPHY (Concluded)

28. Saaty, T. L., "On Jockeying, Collusion, Scheduling, Optimization and Graph Theoretic Queues," Office of Naval Research, Washington, 1961.

29. Sespaniak, L. J., "An Application of Queueing Theory for Determining Manpower Requirements for an In-line Assembly Inspection Department," Journal of Industrial Engineering, vol. 4, pp. 265-267, July-August, 1959.